uc3m | Universidad Carlos III de Madrid

Grado Ingeniería de Sistemas Audiovisuales
2017-2018

Trabajo Fin de Grado

Diseño y desarrollo de proyectos con ANDROID THINGS

David González Ramos

Tutores
Mª Celeste Campo Vázquez
Carlos García Rubio

Leganés, Septiembre 2018

TÍTULO: *DISEÑO Y DESARROLLO DE PROYECTOS CON ANDROID THINGS*

AUTOR: *DAVID GONZÁLEZ RAMOS*

TUTORES: *Mª CELESTE CAMPO VÁZQUEZ*

CARLOS GARCIA RUBIO

EL TRIBUNAL

PRESIDENTE: Luis Sánchez Fernández

SECRETARIO: David Ramírez García

VOCAL: Cristina Brandle Cerqueira

"If one is master of one thing and understands
one thing well, one has at the same time,
insight into and understanding of many things."

Vicent Van Gogh

Agradecimientos

Quiero dejar por escrito mi agradecimiento a todas las personas que, de forma directa u indirecta, han formado parte en la realización de este proyecto.
Para comenzar a mis tutores, por hacerme ver con su asignatura la infinidad de desarrollos posibles y por su apoyo a lo largo del TFG.
A mi familia, por su apoyo incondicional y por aguantar todos los momentos difíciles a lo largo de la carrera.
A mi pareja, por ser mi principal fuente de apoyo.
A mis magníficos compañeros de universidad, por ayudarme en tantas asignaturas y darme la motivación necesaria para terminar.
A la música.

Muchas gracias.

Resumen

Este proyecto se basa en el estudio de la nueva plataforma para el Internet de las Cosas de Google llamada Android Things. Esta plataforma consiste en un sistema operativo basado en Android que permite la construcción de dispositivos inteligentes y conectados para una amplia variedad de aplicaciones de consumo.

El objetivo de este proyecto es el desarrollo y el diseño de algunas aplicaciones que permitan validar el potencial de este sistema operativo. Para llevar a cabo el desarrollo de estas aplicaciones se hará uso de un Hardware compatible con Android Things (*Raspberrry Pi 3 Starter Kit*). Este, consiste en un Kit de desarrollador compuesto por una *Raspberry Pi*, plataforma de desarrollo, junto con una *Rainbow HAT* con numerosos periféricos adicionales.

El desarrollo y el diseño de estas aplicaciones se divide en dos partes donde, en cada una de ellas, se realiza una integración de Android Things con otras tecnologías. En la primera de ellas se estudia otra plataforma de desarrollo llamada Firebase donde se realiza su integración con Android Things con el objetivo de controlar varios periféricos de la *Rainbow HAT* de forma remota en tiempo real. En la segunda parte se hace uso de la biblioteca de inferencia de TensorFlow Lite para Android y se demuestra cómo ejecutarla en Android Things a través de un módulo de cámara capturando y clasificando localmente distintas imágenes.

Para la realización de las distintas aplicaciones que forman este proyecto se hace uso del lenguaje nativo que usa Android, Java. Este lenguaje nos permite aprovechar todo el potencial de los dispositivos tanto a nivel de hardware como de software.

Palabras clave

Android, IoT, Internet de las Cosas, Android Things, Raspberry Pi, Firebase, Tensorflow, Inteligencia Artificial, Visión Cognitiva, RealTime Database.

Abstract

This project is based on the study of the new platform to the Internet of Things from Google called Android Things. The platform involves an Android-based operating system that allows the construction of intelligent and connected devices for a wide variety of applications of consume.

The objective of this project is the development and design of some applications that allows the validation of the potential of this operating system. To carry out the development of these applications we will use a Hardware compatible with Android Things (Raspberrry Pi 3 Starter Kit). it has a Developer Kit consisting of a Raspberry Pi development platform, along with a Rainbow HAT with numerous additional peripherals.

The development and design of these applications are divided into two parts, in each one of them, we integrate Android Things with other technologies. At first another development platform called Firebase is studied, where its integration with Android Things is done in order to control remotely several peripherals of the Rainbow HAT in real time. The second phase uses the TensorFlow Lite inference library for Android and demonstrates how to run it in Android Things through a camera module capturing and classifying locally different images.

For the realization of the different applications that make up this project, we apply the native language used by Android, Java. This language allows us to take advantage of the full potential of the devices, at every level of the hardware and software components.

Keywords

Android, IoT, Internet of Things, Android Things, Raspberry Pi, Firebase, Tensorflow, Artificial Intelligence, Vision Cognitive, RealTime Database.

Índice General

Índice de Figuras

Índice de Código

Índice de Tablas

Capítulo 1

Introducción y objetivos

1.1 Introducción

Internet se puede considerar como un elemento clave en el desarrollo y la historia reciente del ser humano. Su evolución, desde su aparición con sus rudimentarios modéms de 56 KB hasta ahora con sus avanzadas líneas de fibras ópticas, ha sido veloz y sorprendente superando cualquier previsión. [1]

En 1999 Kevin Ashton, impartió una conferencia en Procter & Gamble donde habló por primera vez del concepto de Internet de las Cosas [2]. La revolución de este término vino durante el inicio del siglo XXI con la popularización de conectividad inalámbrica, ya fuese celular o WiFi, que propulsó los objetos conectados. Este crecimiento ha resaltado en esta última década, donde han surgido nuevos conceptos como el WSN (Wireless Sensor Networks) o M2M(Machine to machine), dando lugar al *Internet of Things*.

Se puede definir IoT (*Internet of Things*), como un término nacido en el MIT (*Massachusetts Institute of Technology*) con el cual se define la interconexión tanto de dispositivos como de objetos o personas a través de una red global. Esta forma de conectar cualquier cosa a internet y con ello la creación de infinidad de aplicaciones, ha supuesto una nueva revolución en el mundo de la tecnología. Este concepto no solo ha dotado de conectividad a los objetos, sino que ha creado plataformas para que estos se comuniquen

entre sí y con los humanos para gestionar el gran volumen de datos que estas Cosas de Internet generan. [3]

Tanto las grandes empresas como las pequeñas y medianas empresas (PYMES) apuntan a ser favorecidas por el IoT de tal forma que durante estos años será uno de los ejes en los que se trabajará por parte de compañías. *Internet of Things* está transformando la forma en que estas empresas y los consumidores de todo el mundo hacen su día a día. La tecnología que subyace de este concepto está evolucionando rápidamente

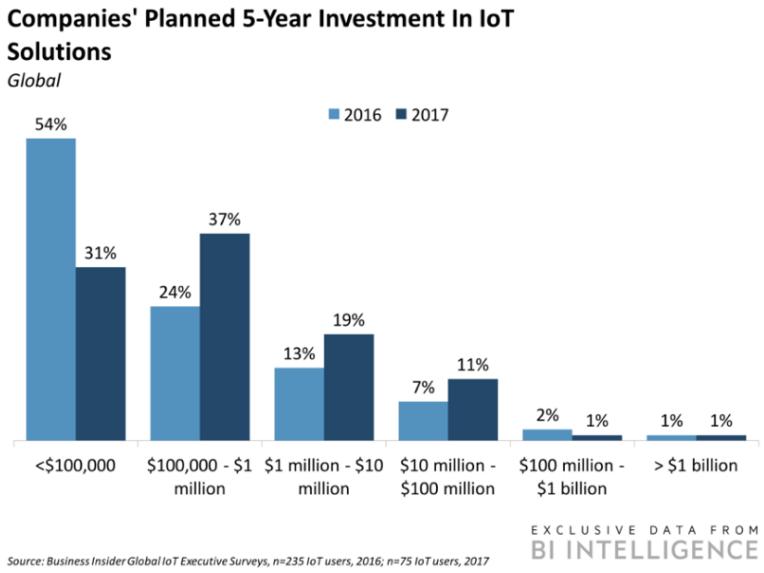


Figura 1: Inversiones planificadas por empresas en tecnologías y soluciones de IoT [1]

Algunos datos según el *IoT Report* realizado por "*Business Insider*" son los siguientes:

Habrá más de 55 mil millones de dispositivos de IoT para 2025, frente a los mil millones en 2017 y casi 5 billones de dólares en inversión agregada de IoT entre 2017 y 2025.

Esta tecnología también tiene una cara negativa. Con su crecimiento, el volumen de datos y de información se ha incrementado de manera exponencial y al mismo tiempo se han multiplicado las vulnerabilidades y peligros innatos a cualquier nueva tecnología. A ello también ha contribuido el hecho de que los fabricantes que han ido implementando el Internet de las cosas no hayan utilizado un estándar, sino que cada uno ha empleado diferentes protocolos de comunicación, sistemas, etc.

1 https://www.businessinsider.com/internet-of-things-report?IR=T

Frente a este problema, las grandes compañías han creado nuevas plataformas para intentar estandarizar estas tecnologías. En el caso de Google con su *Android Things* o Apple con su *Home Kit*, donde han establecido unos mínimos de compatibilidad y seguridad para tener un mayor control de las distintas amenazas y vulnerabilidades.

Actualmente, las grandes empresas de desarrollo de software como Google, Apple, Microsoft, Samsung, LG, etc, están ganando gran parte del territorio del mercado global. Por este motivo Google incorpora al mercado la plataforma de *Android Things* ofreciendo un amplio abanico de oportunidades que posibilita a los desarrolladores a construir e interconectar gran diversidad de objetos versátiles, seguros, de calidad y de relativo bajo costo.

A lo largo de este proyecto se realiza un estudio de esta nueva plataforma de Google, Android Things, a través de la combinación con otras tecnologías en distintos desarrollos. También se valora su potencial con su comparación con otras plataformas y estudiando la diversidad de posibilidades que ofrece.

El Internet de las Cosas es considerado como un punto de inflexión debido a su influencia en distintos aspectos como el acceso a la conexión en lugares remotos, la creación de nuevos modelos de negocios y la gestión de datos. Cada vez es más frecuente encontrarse con nuevos dispositivos capaces de conectarse a Internet y permitir al usuario su control y manejo de forma remota desde cualquier parte del mundo, aunque esto no ha hecho más que comenzar.

1.1.1 Domótica

En la actualidad, existe la posibilidad de instalar sistemas domóticos con numerosas funciones dentro de nuestro hogar, algunas de ellas son el control de luz, el control de temperatura, el control remoto de sistemas de audio y vídeo, entre muchas otras. Poco a poco nuestras casas se están transformando en lo que se denomina hogares inteligentes.

El incremento continuo del precio energético ha aumentado de forma exponencial la demanda de los sistemas de automatización de viviendas, esto implica al mismo tiempo el crecimiento del IoT, es decir, de número de dispositivos interconectados a través de Internet.

La domótica y el IoT son conceptos distintos. Por un lado dispositivos IoT buscan la comunicación de distintos elementos electrónicos, sensores o actuadores, mientras que el objetivo de la domótica es la automatización de respuestas para facilitar y mejorar la vida de las personas. El trabajo conjunto de ambas tecnologías permite simplificarnos la vida de múltiples maneras ofreciendo un sinfín de ventajas.

1.3 Motivación

A lo largo de mis últimos años de carrera y mientras realizaba prácticas en distintas empresas dedicadas a distintas áreas, me percaté tanto del gran potencial que tenían las tecnologías IoT dentro del mercado y de la vida de las personas, como de la necesidad de establecer un "estándar" con el que poder conectar estas tecnologías. Esta es una de las motivaciones por la que se decide realizar este proyecto.

Posteriormente, al reunirme con Mª Celeste Campo y proponerme la idea de trabajar con un nuevo sistema operativo del internet de las cosas, decidí elegir esta idea para mi proyecto. La idea me pareció muy interesante no solo por las posibilidades de esta nueva tecnología y de las numerosas aplicaciones prácticas que tendría dicho desarrollo, sino también por la oportunidad de expandir mi conocimiento de desarrollador en Android hacia un mundo hasta ahora casi desconocido para mí.

Por otro lado, este trabajo Final de Grado surge de la necesidad de realizar investigaciones relacionadas con las posibilidades que nos presentan las nuevas tecnologías emergentes como el IoT y la inteligencia artificial y de su integración en nuestras vidas.

1.2 Objetivos

Este proyecto tiene dos objetivos principales que se unen en uno solo. Por un lado, como se ha mencionado anteriormente, es el diseño y desarrollo de dos proyectos basados en la plataforma de Google Android Things. Otro de los objetivos generales es la introducción al estudio y al desarrollo de servicios proporcionados por otras tecnologías habituales de Google. En concreto se estudia el servicio de *Realtime Database* proporcionado por *Firebase* [4] y una solución liviana de *TensorFlow*[5] para dispositivos móviles e integrados, *TensorFlow Lite*. Esta última consiste en una herramienta de *machine learning* conocida gracias a su eficiencia trabajando con redes neuronales y procesos distribuidos.

Ambos propósitos tendrán como objetico servir como base para desarrollos y mejoras futuras para el diseño de aplicaciones que permitan satisfacer las exigencias de los consumidores.

Para la búsqueda de estos objetivos principales del proyecto se han marcado varios objetivos parciales necesarios para alcanzar el propósito general. Estos se ven reflejados en todo el desarrollo del proyecto.

- Realizar una investigación del Internet de las cosas y en concreto de la plataforma de Android Things.

- Aprender a manejar el software que se ejecuta en el dispositivo con Android Things.

- Comprender y manejar el entorno de desarrollo que proporciona Android Things en uno de sus diseños de hardware. En este proyecto se utiliza el *Raspberry Pi 3 Starter Kit.*

- Avanzar en el aprendizaje de la programación en el lenguaje de Android, aprender a manejar los distintos archivos *Gradle* de un proyecto e implementar distintas dependencias en ellos.

- Aumentar el manejo del principal entorno de desarrollo de Android, Android Studio.

1.4 Estructura de la memoria

Para que la lectura de la memoria se puede realizar de forma sencilla, en este apartado se recogerán los puntos mas importantes del proyecto junto con un breve resumen explicativo.

En el capítulo 1 se realiza una pequeña introducción al marco tecnológico del Internet de las Cosas, seguida de los objetivos y motivaciones por las que se ha llevado a cabo el desarrollo de este proyecto.

En el capítulo 2 se hace un Análisis del Estado del Arte diferenciando distintas secciones donde se abordaran tanto los conceptos mas generales de cada tecnología, hasta las partes más concretas de cada una de ellas utilizadas en el diseño de las distintas aplicaciones. Se comienza con una introducción a los sistemas operativos de *Android* y *Android Things*, seguida de sus arquitecturas y algunos principios sobre el desarrollo de aplicaciones. A continuación se continua con una sección similar para el *Raspberry Pi 3 Starter Kit* hablando sobre su arquitectura y periféricos utilizables. Por último se incluye el marco regulador, donde se realiza un análisis de las regulaciones aplicables al proyecto.

En el capítulo 3 se describe un paso fundamental para el comienzo de los desarrollo, la configuración de la Raspberry. Se incluye tanto la instalación del sistema operativo como los comandos necesarios para su conexión.

El capítulo 4 continúa con la descripción del diseño y desarrollo de los distintos proyectos basados en *Android Things* y del manejo de las dependencias necesarias para su implementación con las tecnologías de *Firebase* y *TensorFlow*.

En el capítulo 5 incluye una planificación de las distintas fases de desarrollo llevadas a cabo durante el desarrollo del proyecto. También se describe el entorno socio-económico del proyecto, el cual recoge un análisis económico de los distintos costes asumidos en la implementación del mismo, seguido de una lista de herramientas utilizadas en dicho desarrollo.

Finalmente, en el capítulo 6 se recogen varias conclusiones finales y se discute sobre el futuro de las distintas tecnologías. Por último se indica las distintas mejoras para la continuación del desarrollo de las distintas aplicaciones detallando algunos pasos para ello.

En la bibliografía se listan las distintas investigaciones y fuentes consultadas para la realización del proyecto. Para su realización se ha seguido la regla *IEEE v9* definida en la Guía temática sobre citas bibliográficas de la Carlos III.

Anexo I. Glosario. Este anexo recoge distintas palabras técnicas utilizadas durante este proyecto.

Anexo II. Manual de administración de productos Android Things. Este anexo incluye un manual orientado al usuario para la creación de productos a través de *Android Things Console.*

Anexo III. Diagrama de Gantt. En este anexo se incluye un diagrama de Gantt con las distintas fases que se han llevado a cabo en el desarrollo del proyecto junto con su duración.

Anexo IV. Función inicialización de la cámara. En este anexo se expone la función destinada a la inicialización previa al uso del periférico de la cámara.

Anexo V. Agregaciones. Este último anexo recoge varias funciones implementadas en los distintos desarrollos.

Anexo VI. *Extended Abstract*. Este anexo recoge, cumpliendo con la normativa vigente para los alumnos del plan 2011, un resumen del proyecto en inglés.

Capítulo 2

Análisis del estado del Arte

2.1 El Sistema Operativo Android

Inicialmente, este sistema operativo fue desarrollado por Android Inc., empresa respaldada por Google y que luego, en 2005, adquirió. Android fue presentado en 2007 junto la fundación del *Open Handset Alliance*, alianza comercial de 35 compañías de hardware, software y telecomunicaciones, para avanzar en los estándares abiertos de los dispositivos móviles. El primer móvil que salió con el sistema operativo Android fue el *HTC Dream* en octubre de 2008. [6]

Android es un sistema operativo de código abierto basado en el *kernel* de Linux para teléfonos móviles lo que permite su utilización sin costes adicionales. Su estructura es simple y es adaptable para cualquier tipo de hardware, fue diseñado principalmente para dispositivos móviles con pantalla táctil, como Smartphones, Tablets, Smartwatch, SmartTv, automóviles, electrodomésticos y una amplia variedad de productos. Android Mobile permite implementar y desarrollar potentes aplicaciones para diferentes tipos de dispositivos de una forma sencilla. [7]

Como resumen de esta plataforma podemos destacar que la portabilidad segura que ofrece es dado a que las aplicaciones son desarrolladas en lenguaje Java, por lo que permite que las aplicaciones sean ejecutadas en cualquier dispositivo. Actualmente existe un nuevo lenguaje compatible con Android llamado Kotlin [8] que funciona sin inconvenientes con Java. Este es compatible con Android Studio 3.0 y versiones posteriores.

Por otro lado, Android también incorpora distintos servicios, como por ejemplo la localización basada en GPS, bases de datos con SQL, síntesis de voz, multimedia, etc. En cuanto a la seguridad de Android se basa en los permisos que otorgue cada usuario.

Cabe destacar que la cuota de ventas en el mercado mundial del sistema operativo móvil Android ha aumentado en lo últimos 7 años del 22 al 85%[2]

2.1.1 Versiones de Android

Desde que la primera versión de Android vio la luz a finales de 2008 ha sido grande la evolución del proyecto a lo largo de las ocho grandes versiones de Android, cuya plataforma a día de hoy mantiene la corona del sistema operativo para móviles más usado en el mundo.

Todas las versiones de Android reciben el nombre de diferentes postres, siguiendo un orden alfabético. A continuación se detallan algunas de estas versiones:

- Android 1.0. Google decidió bautizar a esta primera edición como Apple Pie, fue la que iba en el HTC Dream. Posteriormente están Android 1.1 Banana Bread, Android 1.5 Cupcake y Android 1.6 Donut.
- Android 2.0 y 2.1 Éclair. Introdujo, entre otras cosas, una renovada pantalla de bloqueo. Posteriormente están Android 2.2 Froyo, Android 2.3 Gingerbread.
- Android 3.0 Honeycomb. Fue la primera y la única en ser exclusiva de tablets.
- Android 4.0 Ice Cream Sandwich. Introdujo mejoras como acceder directamente a las aplicaciones desde la pantalla de bloqueo, reconocimiento facial para el desbloqueo o soporte para grabar en 1080p.
- Android 4.1 y 4.2 Jelly Bean. Los añadidos mas importantes fueron Project Butter y Google Now.
- Android 4.4 KitKat. Lo mas destacable fue reducción en el uso de memoria RAM y la llegada de Android Runtime.
- Android 5 Lollipop. introdujo la interfaz Material Design de Google, considerado el punto de inflexión más grande de la historia de Android.
- Android 6 Marshmallow. Mejora la gestión de permisos.
- Android 7 Nougat. Google continuaba perfeccionando Material Design, a la vez que se incorporaban nuevas funcionalidades en el sistema.

2 https://computerhoy.com/reportajes/industria/android-vs-iphone-guerra-smartphones-cifras-271447

- Android 8 Oreo. Fue presentada en agosto de 2017 y su añadido mas importante fue Project Treble. Consiste en un cambio de estructura cuyo objetivo es el de modularizar el propio sistema operativo. Más tarde, con Android 8.1 llega la API para redes neuronales, que permite a los desarrolladores implementar sistemas de IA y Machine Learning en sus apps.
- Android 9.0 Pie. En marzo de 2018, Google lanza la primera versión preliminar para desarrolladores donde incorpora una mayor flexibilidad a la hora de desarrollar y diseñar sus apps, mejoras en el rendimiento gracias a ART y de ahorro de energía a través de DOZE, etc.

Version	Codename	API	Distribution
2.3.3 - 2.3.7	Gingerbread	10	0.2%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	0.3%
4.1.x	Jelly Bean	16	1.2%
4.2.x		17	1.9%
4.3		18	0.5%
4.4	KitKat	19	9.1%
5.0	Lollipop	21	4.2%
5.1		22	16.2%
6.0	Marshmallow	23	23.5%
7.0	Nougat	24	21.2%
7.1		25	9.6%
8.0	Oreo	26	10.1%
8.1		27	2.0%

Figura 2. Distribución de dispositivos Android según la versión del sistema [3]

Actualmente conviven muchas de las versiones mencionadas en el mercado, esto se debe a que muchos terminales no se han actualizado a nuevas versiones. En la tabla anterior se representa el porcentaje de terminales que usan las diferentes versiones.

2.1.2 Arquitectura de Android

La complejidad de la estructura de las aplicaciones móviles desarrolladas en Android se debe a que Android se basa en una arquitectura de componentes. Esta arquitectura consiste en un conjunto de librerías de Android con el objetivo de mantener y robustecer las aplicaciones.

3 https://developer.android.com/about/dashboards/?hl=es-419

Esta arquitectura de componentes con la que cuenta Android se puede dividir en las siguientes capas [9]:

- **Kernel de Linux**. Este núcleo incluye los drivers necesarios para el funcionamiento del dispositivo. También proporciona servicios de seguridad, manejo de memoria, multiproceso y soporte para controladores de dispositivo. Esta capa no es una distribución de Linux, sino que los desarrolladores pueden modificarla para adaptarla a las necesidades propias de Android.

- **Capa de abstracción del hardware**. Brinda interfaces estándares que exponen las capacidades de hardware del dispositivo al framework de la Java API de nivel más alto.

- **Librerías**. Son librerías en C/C++ cuyo objetivo es hacer ciertas operaciones mas sencillas para el desarrollador y ayudar a mejorar el rendimiento. Muchas de estas utilizan código donde se pueden destacar WebKit o SQLite.

- **Android Runtime**. Posteriormente se destaca el entorno de ejecución basado en la máquina virtual de Java. Desde Android 5.0 se utiliza, sustituyendo a Dalvik, Android Runtime (ART), que usa compilación *ahead of time* (AOT).

- **Framework de aplicaciones**. Todo el conjunto de funciones del SO Android está disponible mediante API escritas en el lenguaje Java.

- **Aplicaciones**. En esta capa superior está formada por un conjunto de aplicaciones centrales como el calendario, SMS, el correo electrónico, contactos, navegación en Internet, entre otras.

2.1.3 Introducción a la programación de Android

Una aplicación móvil está formada por una serie de componentes principales de software: actividades, servicios, fragmentos, proveedores de contenido (*content providers*) y receptores de emisión (*broadcast receivers*). Otros elementos importantes dentro de una aplicación móvil son los *intents* y los *widgets*.

- **Activity**. Componente principal de la interfaz gráfica que contiene una pantalla con la que los usuarios pueden interactuar para realizar una acción. Cada *activity* tiene asociada una vista.
- **Views**. Componentes básicas para la construcción de la interfaz gráfica a través de controles básicos proporcionados por Android (botones, cuadros de texto, etc)
- **Service.** Componente que no lleva asociada interfaz gráfica cuya función es ejecutarse en segundo plano.
- **Content Provider.** Componente que permite compartir datos entre aplicaciones utilizando mecanismos de seguridad.

- **Broadcast Receiver.** Componente que permite recibir y reaccionar a notificaciones generadas por otras aplicaciones o por el sistema.
- **Intent.** Elementos básicos que permiten solicitar acciones a otros componentes de la aplicación. Se utilizan para comenzar una actividad, para inicial un servicio, para entregar un mensaje, etc.
- **Widget**. Elemento visual que muestra información en la pantalla, generalmente interactivo, con el objetivo de conocer o usar la aplicación sin necesidad de abrirla.

Una aplicación consiste generalmente en múltiples actividades conectadas entre sí. [9] Los métodos *callback* sirven para notificar los cambios de estado del ciclo de vida de una actividad. Existen diferentes métodos *callback* que podría recibir una actividad como consecuencia de un cambio de estado y cada uno de ellos ofrece la oportunidad de realizar una tarea específica. Algunos de estos métodos son los siguientes:

- onCreate(): Se llama al crear una actividad y sirve inicializar los componentes fundamentales de tu actividad.
- onStart(): Se llama justo antes de que la actividad se vuelva visible.
- onPause():Se llama cuando el sistema está a punto de reanudar otra actividad.
- onResume():Se llama cuando la actividad va a empezar a interactuar con el usuario después de estar en un estado de pausa.
- onStop(): Se llama cuando la actividad ya no es visible para el usuario.
- onRestart():Se llama después de que una actividad se haya detenido y antes de que se inicie de nuevo.
- onDestroy():Se llama antes de que se destruya la actividad.

Todas las aplicaciones desarrolladas en Android cuentan en su directorio con un archivo llamado *AndroidManifest.xml*, el cuál es el encargado de proporcionar información esencial de la aplicación necesaria para su ejecución.

En cuanto al entorno de desarrollo oficial utilizado para el desarrollo de aplicaciones en Android se utilizada Android Studio. Este ofrece numerosas funciones a lo largo de la compilación de las aplicaciones, algunas de estas son la emulación veloz, la compilación basada en Gradle y la gran cantidad de herramientas y frameworks de prueba.

En la sección 2.2 se realiza un estudio del sistema operativo de Android Things donde se resaltan las diferencias más destacables con los desarrollos en Android.

2.2 Android Things

2.2.1 Introducción

Cada pocos años alguna empresa presentaba su propia solución para intentar hacer accesible al consumidor medio ese hogar y esa vida conectada a trasvés de la tecnología del Internet de las cosas, pero luego siempre quedaba estancada por culpa de incompatibilidades con los protocolos de dispositivos de la competencia y por los altos precios.

Google llevaba ya varios años intentando conquistar el hogar conectado y el Internet de las cosas, pero fue en 2016, cuando presenta a Android Things como el nuevo sistema operativo basado en Android para el Internet de las Cosas, el sistema operativo para dispositivos inteligentes.

Esta nueva plataforma se construye sobre el proyecto de Google presentando anteriormente como Project Brillo. Google actualiza este antiguo proyecto, ofreciendo una plataforma apta para todo tipo de objetos conectados. Este incorpora además servicios disponibles en la plataforma de desarrollo de Android, dotando de esta forma de mayor funcionalidad a esta versión para que los desarrolladores puedan crear apps específicas en Android Things.

Google, para la comunicación entre sí de los distintos dispositivos, propone y actualiza el protocolo de comunicación *Weave*, protocolo de conexión que utilizaba Project Brillo. Así los dispositivos podrán usar una única API de comunicación para que puedan interactuar entre ellos de forma sencilla, y sin problemas de compatibilidad por usar distintos sistemas operativos. Este protocolo de comunicación de Google tiene muy buena acogida por el sector de domótica.

Este sistema operativo y plataforma de desarrollo no está diseñado para ejecutarse sobre *Smartphones* sino para dispositivos inteligentes que se van a conectar a Internet como bombillas conectadas, Raspberry Pi, routers, termostatos, etc.

Desde su inicio como versión para desarrolladores en 2016, esta plataforma ha estado en formato prueba durante estos dos años. Durante este periodo, esta compañía tecnológica trabajó de cerca con nuevos productos de las marcas LG, Lenovo, etc, las cuales han incorporado esta nueva plataforma a su sistema. El Internet de las cosas tiene en Android Things a un sistema ligero, con experiencia, pleno de posibilidades y listo para adaptarse a cualquier dispositivo.

En cuanto a la programación en Android Things es bastante similar al desarrollo de aplicaciones en Android. Los proyectos se realizan en el mismo entorno de desarrollo, Android Studio y la estructura de los proyectos es la misma, con su *AndroidManifest.xml,* sus archivos *gradle*, declaración de funciones y declaración de clases y actividades.

En cuanto a las diferencias mas generales de estos dos sistemas operativos se definen en la sección 2.2.3. Por otro lado, algunos de los conocimientos adquiridos necesarios para la realización de los distintos desarrollos relacionados con estas diferencias son:

- Ahora con Android Things la compilación de los proyectos no se realiza sobre un dispositivo móvil, sino sobre otro tipo de hardware compatible sin interfaz visual. Por este motivo en los desarrollos de Android no se asocian vistas a las actividades.
- Ahora en los desarrollos de Android Things la lógica que sirve de base reside en las funciones para inicializar y destruir los distintos periféricos usados en cada proyecto.

Android Things es una plataforma que extiende de Android Mobile que facilita a los desarrolladores la creación y la integración del ecosistema de desarrollo de Android incluyendo APIs y herramientas de desarrollo y a los fabricantes la creación de dispositivos IoT. Dentro de esta plataforma podemos distinguir cuatro partes importantes [11]:

- Su sistema operativo como variante de Android
- Arquitectura
- Amplio Hardware
- Consola de desarrollador para administrar dispositivos y enviar actualizaciones.

2.2.2 Sistema operativo Android Things

El software por el que está compuesto el sistema operativo de Android Things, permite construir aplicaciones en distintos dispositivos utilizando algunos servicios y componentes extendidos de Android Mobile como el SDK de Android y los servicios de Google Play. Gracias a esto, estas aplicaciones se pueden integran fácilmente con otros servicios de Google, como TensorFlow (software de Google orientado a problemas de Deep Learning), Firebase (plataforma de desarrollo móvil en la nube) y Google Cloud Platform [12] (aplicación que permite crear e implementar aplicaciones, sitios web y servicios en la misma infraestructura que Google) utilizando la gran variedad de bibliotecas de desarrollador de Android Mobile.

Esta plataforma no posee ninguna aplicación de usuario que sirva de navegador para su configuración o desarrollo, sino que este sistema operativo está diseñado para comenzar a trabajar directamente con el dispositivo en concreto donde se ejecuten las aplicaciones que se han creado.

Para el desarrollo de Android Things se utiliza el mismo lenguaje que para implementar en Android, aún así se destacan pequeñas diferencias:

- Un ajuste de la plataforma para reducir los tiempos de arranque como el consumo de memoria.
- La agregación de nuevas APIs con el objetivo de una adecuada integración con el hardware específico.

El objetivo general de Google es proporcionar a los fabricantes un sistema operativo administrado y hardware certificado que permita la despreocupación por el sistema y su mantenimiento para promover el diseño y el desarrollo de diferentes productos.

2.2.3 Arquitectura Android Things

Como se mencionaba, gracias a las APIs que proporciona la biblioteca de soporte de Things, Android Things no está limitado a la integración en los dispositivos móviles tradicionales, sino que permite la integración con gran variedad de tipos de Hardware.

Existen algunas diferencias importantes entre el desarrollo de proyectos para Android Things, ya que están optimizados para dispositivos integrados y el desarrollo de aplicaciones en Android Mobile, algunas de ellas son las siguientes:

- Existen distintas funciones y APIs de Android que actualmente no son compatibles con los dispositivos de Android Things.
- Los dispositivos de Android Things están optimizados para el uso de una sola aplicación que se inicia automáticamente al inicio del sistema.
- Estos dispositivos integrados suelen tener limitaciones de memoria, por lo que la plataforma requiere que las aplicaciones mantengan las bibliotecas nativas dentro del APK en tiempo de ejecución.
- Los dispositivos de Android Things tiene un acceso más flexible a periféricos de hardware y controladores.

2.2.4 Actualizaciones Android Things

Con el objetivo de que el usuario se centre únicamente en el desarrollo y construcción de aplicaciones y al mismo tiempo garantizar su protección en todo momento, Google ofrece las distintas actualizaciones y diferentes parches de seguridad del sistema operativo.

Uno de los principios básicos de Android Things es impulsar dispositivos que permanecen seguros a lo largo del tiempo y una de las técnicas para conseguirlo es proporcionar actualizaciones de software por aire (OTA). Estas actualizaciones están diseñadas para evitar una actualización corrupta, restaurando una versión anterior estable del sistema al detectar algún error en cualquier actualización.

Las aplicaciones de Android Things usan la misma estructura segura para el envío de actualizaciones a través de Internet que las aplicaciones comúnmente diseñadas para teléfonos y tablets. La instalación de estas aplicaciones, se realizan desde la consola de Android Things, por lo que Android Things no incorpora Google Play Store.

Android Things Console proporciona una implementación fácil y segura de las actualizaciones en sus dispositivos compatibles. Consiste en una infraestructura que permite tanto entregar actualizaciones del sistema por aire a dispositivos de desarrollo y de producción como administrar dichos dispositivos.

Para los fabricantes y los desarrolladores, *Android Things Console* ofrece [13]:

- Descargar e instalar la última imagen del sistema Android Things.
- Crear imágenes de fábrica que contengan aplicaciones OEM junto con la imagen del sistema.
- Lanzar las actualizaciones sin cable (OTA), incluidas las aplicaciones OEM y la imagen del sistema a los dispositivos.
- Monitorizar un análisis informativo de los productos para visualizar su funcionamiento.

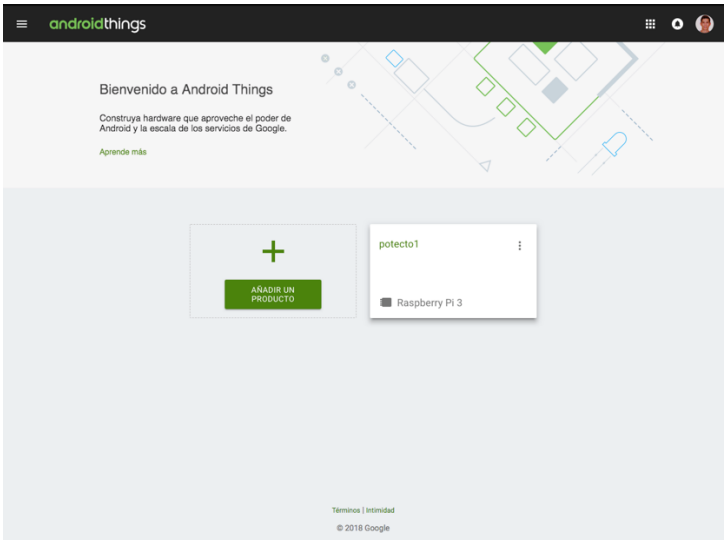


Figura 3. Pantalla de inicial de la consola de Android Things [4]

4 Pantalla de inicio *Android Things Console*

Google permite a los usuarios no comerciales la administración de hasta 100 dispositivos en *Android Things Console* para trabajar en el desarrollo de sus productos. Una vez que superen los 100 dispositivos o deseen lanzar un producto comercial, deberán firmar un acuerdo.

Esta administración de productos de Android Things es necesaria para llevar cabo cualquier desarrollo con esta tecnología. La administración de un producto se divide en su creación y su configuración. En el anexo I se adjunta un manual de usuario con una breve guía para la realización de estos pasos.

2.2.5 Android Things 1.0

A pesar de que durante este tiempo el estado de Android Things ha sido de versión preliminar, diferentes fabricantes han trabajado con estos objetos conectados obteniendo productos como altavoces inteligentes (LG ThinQ) y pantallas conectadas (Lenovo) gracias al soporte nativo para *Google Assistant*. Según Google, esta versión preliminar de Android Things tuvo más de 10.000 descargas de kits de desarrollo de software (SDK) y más de 10.000 desarrolladores realizaron comentarios. Ahora esta tecnología será utilizada por mas fabricantes y será integrada a muchos más dispositivos aprovechando que por fin tiene una versión 100 % estable.

El 7 de mayo de este año la compañía anunció el lanzamiento de su plataforma Android Things 1.0. Esta es la primera versión estable de su sistema operativo cuyo objetivo es conquistar nuestro hogar conectado con los dispositivos inteligentes.

Google se ha comprometido a crear parches de seguridad por tres años con la posibilidad de extender el periodo de soporte, lo que hace que sea más fácil llevar prototipos al mercado. Estas actualizaciones de software se realizarán mediante servidores OTA.

Android Things 1.0 incluye soporte para sistemas de producción en módulos (SoM) basados en las siguientes plataformas de hardware: NXP i.MX8M, Qualcomm SDA212, Qualcomm SDA624 y MediaTek MT8516. Esta versión está optimizada para consumir muy pocos recursos funciona bajo estos sistemas en módulos. Las Raspberry Pi 3 Modelo B y NXP i.MX7D serán siendo compatibles pero solo para prototipos, no para productos comerciales, mientras que los dispositivos NXP i.MX6UL no continuarán con soporte.

Android Things 1.0 presenta una variedad de nuevas funciones y capacidades para usuarios y desarrolladores con las que Google espera expandirse rápidamente en el mercado tecnológico.

2.3 Hardware

2.3.1 Sensores

Los sensores y actuadores están cobrando un papel muy importante dentro de este tipo de tecnologías. Un sensor es un dispositivo que detecta una determinada acción externa, temperatura, presión, etc., y la transmite adecuadamente.[14]

Hoy en día, gracias a la reducción del tamaño de sus componentes, este Hardware cada vez tiene más fácil alcanzar su objetivo, comunicar el mundo físico con el virtual. Los sensores son la capa del Internet de las cosas más cercana al usuario que permiten la recogida de información necesaria.

Se puede considerar un sensor a cualquier dispositivo que detecta o mide una magnitud física y entrega una información en forma digital. Algunas de las magnitudes que estas sensores puede recoger son las siguientes:

- Posición, velocidad, desplazamiento. Los dispositivos móviles actuales con sus acelerómetros y giroscopios rastrean nuestros movimientos en cualquier parte.

- Información ambiental. Permiten recoger información ambiental, como temperatura o humedad, dando la posibilidad de controlar el clima en distintas zonas.

- Información biológica y de salud. Como por ejemplo los las pulseras y relojes de actividad que miden el número de pasos que damos, nuestra frecuencia cardíaca o el ritmo respiratorio.

- Conectividad con módulo de Wifi o de GPS. El reto de la tecnología IoT es transformar toda esta cantidad de información capturada por estos sensores en algo productivo. Esto es el principio por el que se lleva trabajando en el mundo tecnológico con técnicas de Big Data y la Inteligencia Artificial.

2.3.2 Hardware Android Things

La creación de dispositivos destinado para el Internet de las cosas suele llevar de la mano numerosos problemas y dificultades:

- La integración del sistema en el dispositivo suele requerir una profunda experiencia técnica.

- La construcción y el despliegue de dispositivos es costoso y lento
- La certificación del dispositivo y la compatibilidad de la red es costosa.

Android Things se enfrenta a estos problemas con la creación de dispositivos con el sistema de producción en módulos (SoM) certificados y placas de desarrollo con un diseño que intenta facilitar su proceso de producción. El hardware Android Things está pre-certificado tanto por diferentes proveedores que proporcionan soporte a largo plazo para cada lanzamiento estable de hardware, como por agencias reguladoras.

Las plataformas de Hardware compatibles con Android Things son las siguientes:

➢ Plataformas de producción

El SoM es totalmente compatible con casos de uso de producción ya que están certificados por Google para cumplir lo requisitos de seguridad.

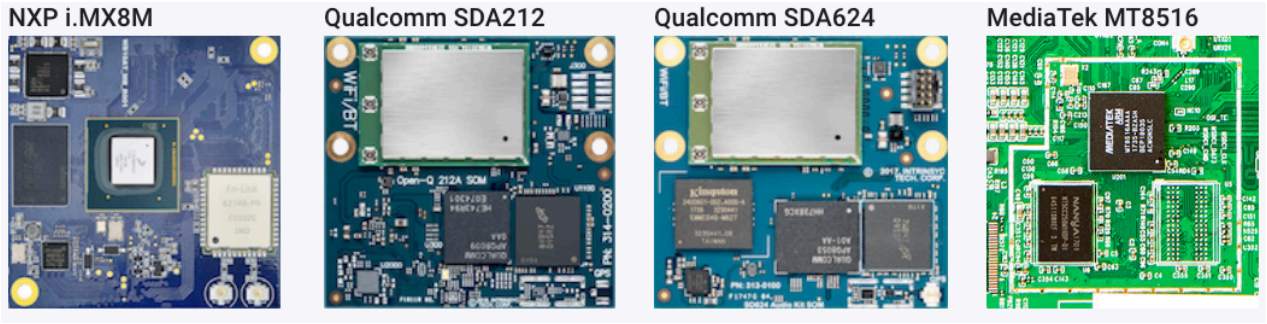


Figura 4. Plataformas de producción [5]

➢ Plataformas de desarrollo:

Por otro lado, Android Things proporciona varios tipos de plataformas de desarrollo para permitir la creación de prototipos y las pruebas. Estas no cumplen con los requisitos de seguridad de Google ya que no están obligadas a recibir actualizaciones de seguridad y estabilidad.

Para el desarrollo de las aplicaciones se ha elegido una plataforma de este tipo, Raspberry Pi Modelo B, conforme al objetivo perseguido por el proyecto de servir como base para desarrollos y mejoras futuras que permitan satisfacer las exigencias de los consumidores.

5 https://developer.android.com/things/hardware/

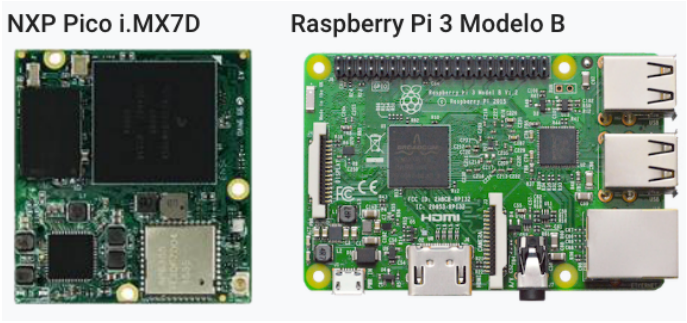


Figura 5. Plataformas de desarrollo [6]

Además de estas plataformas, Android Things proporciona a los fabricantes varios kits de desarrollo de software para desarrollar dispositivos IoT sin necesidad de preocuparse por el mantenimiento.

Figura 6. Raspberry Pi 3 Starter Kit [7]

Figura 7. Kit de inicio NXP i.MX7D [7]

6 https://developer.android.com/things/hardware/
7 https://developer.android.com/things/get-started/kits

Este último kit consiste en un kit de producción, ya está formado por una plataforma de desarrollo denominada NXP i.MX7D, la cual posee la certificación de seguridad para Android Things. En cuanto al primero de ellos, *Raspberry Pi 3 Starter Kit,* será el kit utilizado en el desarrollo del proyecto, cuyo contenido se expone en la sección 2.4.3.

2.3.3 Raspberry Pi 3 Starter Kit

El *Raspberry Pi 3 Starter Kit* es el kit utilizado durante todo el desarrollo del proyecto. Este kit incluye la popular computadora de una sola placa que se ha usado para todo, desde educación hasta robótica. Este kit está destinado para realizar una introducción en el desarrollo de aplicaciones de Android Things en la Raspberry Pi y prototipar diferentes proyectos para su posterior traslado a producción. Este kit está formado por los siguientes componentes:

- Tablero de desarrollo Raspberry Pi 3 Model B (actualmente Android Things no es compatible con Raspberry Pi 3 B +)
- Rainbow HAT
- Protectores de plástico
- Adaptadores de toma de corrientes
- Tarjeta microSD de 16Gb con adaptador
- Fuente de alimentación oficial de Raspberry Pi

En la figura 9 se representan todos estos componentes y la Raspberry Pi montada junto con los protectores de plástico.

Figura 8. *Raspberry Pi 3 Starter Kit*

2.3.3.1 Raspberry Pi 3

Raspberry Pi es un ordenador de placa reducida o ordenador de placa simple de bajo coste creada en el Reino Unido por la Fundación Raspberry Pi, cuyo objetivo fue y sigue siendo la estimulación de la enseñanza de la informática y el desarrollo de pruebas de concepto.

En el año 2016 aparece el modelo: *Raspberry Pi 3 Model B*, que proporciona una CPU ARM Cortex-A53 de cuatro núcleos de 64 bits que funciona a 1,2 GHz, cuatro puertos USB 2.0, redes cableadas e inalámbricas, salida de video compuesto y HDMI, y un conector GPIO de 40 pines para proyectos de interfaz física. En la siguiente imagen se representa un esquema de sus pines GPIO ilustrando su ubicación en la placa junto con los distintos componentes que forma su estructura:

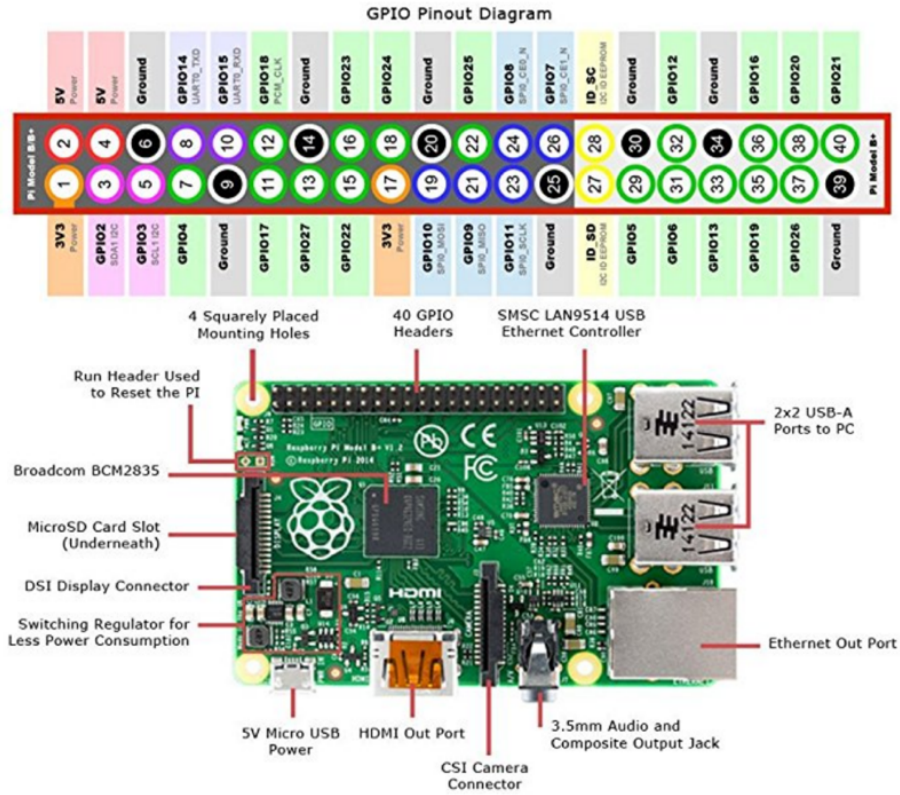


Figura 9. *Raspberry Pi 3 Model B y sus componenetes* [8]

Algunos de estos pines se multiplexan entre varias funciones de la placa, por lo que algunas funciones de ella no se pueden usarse simultáneamente (por ejemplo, habilitar Bluetooth y usar el puerto UART0).

8 https://www.amazon.it/RASPBERRY-PI-Raspberry-Desktop-Computer/dp/B00LPESRUK

2.3.3.2 Rainbow HAT

La *Rainbow HAT* que hay disponible para la Raspberry Pi permite experimentar gracias a una serie de sensores, displays, pulsadores y otros elementos. En la *Rainbow HAT* puede implementarse numerosos desarrollos como por ejemplo una estación meteorológica, un reloj o un temporizador. A continuación se listan las características que incorpora la *Rainbow HAT*:

- Siete LED multicolores APA102
- Cuatro displays de catorce segmentos alfanuméricos en color verde
- Chip controlador HT16K33
- Tres botones táctiles capacitivos
- Chip controlador de los pulsadores capacitivos Atmel QT1070
- LED azul, verde y rojo
- Sensor de presión y temperatura BMP280
- Zumbador piezoeléctrico
- Pins para conexión de servos, I2C, SPI y UART de 3,3 V.

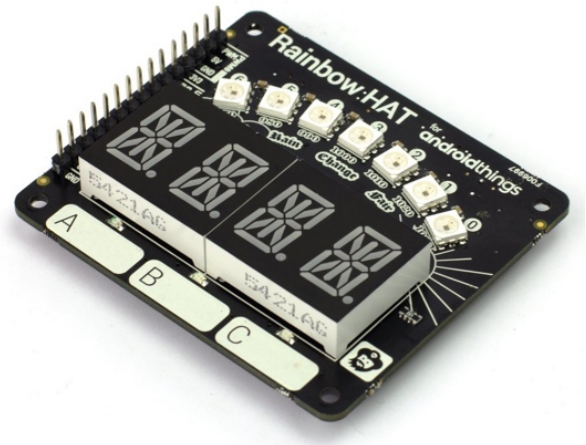


Figura 10. *Rainbow HAT* [9]

Por tanto, este HAT sirve para experimentar con un montón de protocolos en la Raspberry Pi 3 con Android Things y además tener acceso a varios conectores GPIO para otros proyectos. También tiene una API de Python completa para usar en Raspbian.

9 https://www.xatakahome.com/trucos-y-bricolaje-smart/con-este-hat-para-la-raspberry-pi-3-podras-dar-tus-pinitos-con-android-things

2.4 Alternativas a Android Things

Como se menciona anteriormente IoT no consiste en una única pieza tecnológica sino que la forman una combinación de dispositivos, redes y sensores trabajando con un mismo objetivo. La parte encargada de conseguir la unión de estas piezas son las plataformas IoT, con las que gracias a esto permiten la creación de las numerosas aplicaciones que proporciona el IoT.[15]

Uno de los pasos más importantes a la hora de desarrollar cualquiera de sus aplicaciones es la elección de la plataforma IoT. Esta elección marcará tanto el porcentaje del éxito actual como el fututo de la aplicación. En la actualidad existen otras plataformas IoT que compiten en el mercado con la reciente plataforma de Android Things. Algunas de estas son las siguientes:

- Azure IoT Hub[10]. Esta plataforma ofrecida por Microsoft proporciona un conjunto de aplicaciones IoT en azure con el objetivo de la creación de proyectos específicos comenzando desde cero o desde soluciones pre-configuradas. Algunos servicios que ofrece esta plataforma son el servicio de *Stream Analytics* para analizar y procesar los datos, *Web Apps* para trabajar con la parte visual, *Event Hub* para configurar y lanzar eventos y almacenamiento en base de datos ente otros.

- Amazon Web Services IoT (AWS)[11]. Esta plataforma permite la conexión, el procesamiento de datos y la creación de interacciones y aplicaciones con el objetivo de interactuar entre los distintos dispositivos conectados a la nube de Amazon Web Services. Al igual que Google con Android Things, Amazon facilita la integración de esta plataforma con otros servicios AWS. Algunos des estos son Amazon Machine Learning, Amazon Kinesis, AWS CloudTrail, etc.

- IBM Watson IoT[12]. El objetivo de esta plataforma proporcionada por IBM, al igual que las otras es el almacenamiento, la gestión y el análisis de los diferentes datos proporcionados por los dispositivos conectados. Podemos diferenciar que sus servicios se realizan completamente desde la nube permitiendo un contacto con los datos en tiempo real.

10 https://azure.microsoft.com/es-es/services/iot-hub/
11 https://aws.amazon.com/es/iot/
12 https://www.ibm.com/internet-of-things

Otras plataformas que tiene gran relevancia en el mercado de tecnologías IoT actual son *Oracle Inernet of Things Cloud Service, Xively, Samsumg Artik, Adafruit.IO, Carriots, Ubidots, etc.* En la siguiente figura se comparan algunas características de las plataformas IoT más destacadas.

	Microsoft Azure IoT Hub	Amazon AWS IoT	IBM IoT Foundation
Protocols	HTTP, AMQP, MQTT and custom protocols using protocol gateway project)	HTTP, MQTT	HTTP, MQTT
Communication Patterns	Telemetry, Command	Telemetry, Command (state change)	Telemetry, Command
Certified Platforms	Intel, Raspberry Pi 2, Freescale, Texas Instruments, MinnowBoard, BeagleBoard, Seeed, resin.io	Broadcom, Marvell, Renesas, Texas Instruments, Microchip, Intel, Mediatek, Qualcomm, Seeed, BeagleBoard	ARM mbed, Texas Instruments, Intel, Rasppberry Pi, Arduino Uno
SDK / Language	.Net and UWP, Java, C, NodeJS	C, NodeJS	C#, C, Python, Java, NodeJS
Security	TLS (only server authentication)	TLS (mutual authentication)	TLS
Authentication	Per-device with SAS token	X.509 certificate client authentication, IAM service, Cognito service	Per-device with token
Pricing	Paying for IoT Hub unit related to number of devices and messages per days	Paying million messages traffic (published from devices + delivered to devices)	Paying related to number of devices, data traffic and data storage

Figura 11. Comparativa entre plataformas IoT [13]

2.5 Firebase

En sus inicios, Firebase comenzó siendo un simple proyecto de base de datos, pero desde que Google la compró en 2014 la ha ido mejorando con la compra de otros equipos, como *Divshot* (plataforma para web HTML5), hasta convertirla en la plataforma completa que es ahora.

Esta plataforma empezó a darse a conocer gracias al servicio de back-end que ofrece. Este consiste en la implementación sencilla de una base de datos en tiempo real y el acceso a ella a través de sus librerías. Al principio Firebase fue diseñado solo para aplicaciones iOS y Android, pero hoy en día también es compatible con aplicaciones web.

En Firebase podemos clasificar sus características dentro de cuatro grandes pilares cuyo objetivo es un posicionamiento claro dentro del mercado para llamar la atención de aplicaciones de cualquier tipo. Estos cuatro pilares son:

- Analíticas: El servicio de analítica que incluye, *Firebase Analytics*, consiste en una solución gratuita para tener todo tipo de medidas acerca de un usuario dentro de las aplicaciones y poder gestionarlo todo desde un único panel.

13 https://unpocodejava.com/2016/11/02/tabla-comparativa-plataformas-iot-azure-iot-hub-vs-aws-iot-vs-watson-iot-foundation-vs-sofia2-iot-platform/

- Desarrollo: Google ofrece en conjunto de servicios para construir mejores apps, permitiendo delegar en Firebase algunas funciones como el almacenamiento de datos en la nube, reporte de errores o procesos de autentificación.

- Crecimiento: Este servicio recoge una serie de recursos que permiten la captación y la gestión de usuarios de las aplicaciones. Algunos de estos recursos son las notificaciones o invitaciones.

- Monetización: Google ofrece un servicio de monetización a través de la integración de Firebase con *AdMob* (plataforma de publicidad para móviles)

El primer y único requisito para el desarrollo de cualquier proyecto con Firebase es una cuenta de Gmail. Firebase utiliza una base de datos en la nube llamada *NoSQL*, con ella ofrece el servicio de almacenamiento de información en formato JSON y su sincronización con las aplicaciones en tiempo real incluso sin conexión a Internet. Esta base de datos, *NoSQL*, es más conocida como *Firebase Realtime Database*.

La base de datos es también accesible a través de una REST API, la cual permite crear conexiones de HTTP para recibir notificaciones *push* de un servidor.

Figura 12. Firebase RealTime Database [14]

En la actualidad, los problemas de desarrollo del *back-end* y los relacionados con el servidor se están convirtiendo en una gran complicación para los desarrolladores debido a no poder centrar toda su atención a la implementación de las aplicaciones. Por este motivo, Firebase es considerada una de las tecnologías más revolucionarias en el desarrollo en general.

14 https://theengineerscafe.com/tag/firebase-realtime-database-tutorial-android/

2.6 Tensorflow

Durante estos últimos años se han realizado grandes cambios tecnológicos que han producido una revolución en el *Machine Learning*. Gracias a esto, la IA esta cobrando mucha más fuerza y se está aplicando en nuevas áreas, como a Big Data o IoT, afectando a la vida de todos.

El origen de *Tensorflow* tuvo lugar tras muchos años de desarrollo por parte de varios trabajadores de Google que formaban parte del proyecto dedicado a la investigación del aprendizaje automático, Googel Brain Team. Estos investigadores desarrollaron en 2011 *DistBelief*, predecesor cerrado de *TensorFlow*. Hoy en día, esta plataforma, aunque está orientada a la investigación del aprendizaje automático, no solo se centra en este campo, sino que consiste en un sistema lo bastante amplio donde puede aplicarse a muchos otros campos [16].

En 2015, el proyecto de *Googel Brain Team*, libera la librería de *tensorflow* bajo la licencia Apache y fue en febrero de 2017 cuando llegó la versión 1.0 con numerosas mejoras como la integración con otras bibliotecas como Keras.

En la actualidad, *Tensorflow* es una biblioteca de software libre de computación numérica orientado al campo de *Deep Learning* utilizado para el cálculo numérico mediante diagramas de flujo de datos. Esta biblioteca está diseñada para su fácil utilización en distintos sistemas de Hardware, lo que hace que esta tecnología forme parte de muchos avances tecnológicos de los últimos años. *Tensorflow* es una herramienta que proporciona varios métodos de construcción y entrenamiento de redes neuronales de aprendizaje profundo, a través de algoritmos que simplifican la complejidad que supone el manejo de estos métodos.

- Permite realizar distintas tareas desde sus niveles más bajos hasta los más altos a través de los servicios de APIs en todos los niveles:

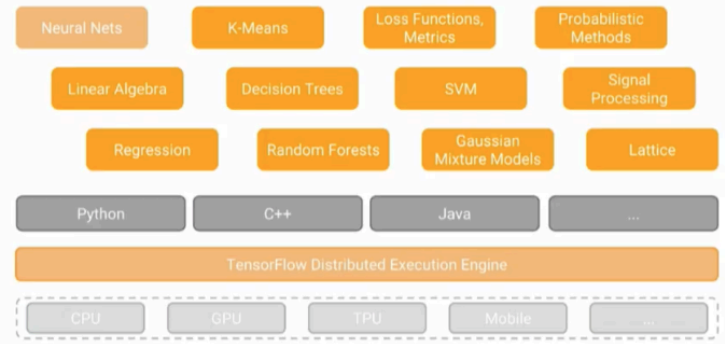


Figura 13. Distribución APIs Tensorflow [15]

15 https://medium.com/ai-learners/tensorflow-dev-summit-resumen-3e77862f5416

- Permite la visualización resultados a través de TensorBoard. TensorBoard es un conjunto de herramientas que permiten la visualización de resultados de ejecuciones relativos a modelos entrenados.

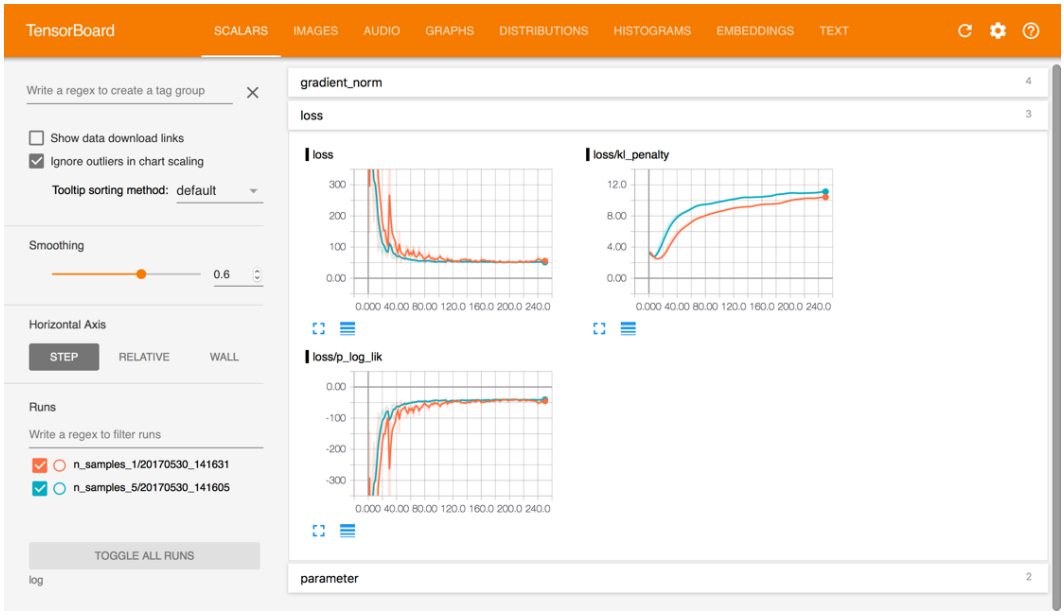


Figura 14. Ejemplo representación de funciones TensorBoard [16]

Las aplicaciones de Tensorflow son infinitas, a parte de estas aplicaciones, esta tecnología se puede extrapolar a cualquier otra área como la medicina o las artes. Google ha sido una de las empresas pioneras en realizar investigaciones en el campo de la Inteligencia Artificial, impulsando su desarrollo constantemente. Con TensorFlow ha conseguido sin duda un salto a la innovación.

2.6.1 TensorFlow Lite para Android

Dentro del mundo de *Machine Learning*, los dispositivos móviles ofrecen muchas posibilidades gracias a su disposición de multitud de sensores y su constante interacción con los usuarios. Por otro lado, trabajar con *Machine Learning* en dispositivos móviles tiene también algunos problemas como la limitación de memoria o del poder computacional.

TensorFlow Lite para Android consiste en una solución liviana de TensorFlow para dispositivos móviles e integrados [17]. TensorFlow Lite utiliza varias técnicas para lograr esta baja latencia, como la optimización de núcleos para aplicaciones móviles o núcleos

16 http://edwardlib.org/tutorials/tensorboard

cuantificados que permiten modelos más pequeños y más rápidos. Estos modelos a su vez perderán parte de la precisión, en la siguiente imagen se comparan la velocidad dos tipos de modelos.

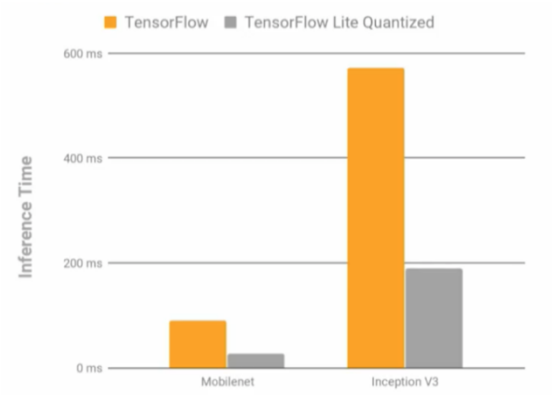


Figura 15. Comparación de velocidades de varios modelos[17]

Para obtener un modelo TensorFlow Lite, TensorFlow nos proporciona la herramienta, convertidor de TensorFlow, para transformar lo modelos capacitados en TensorFlow al formato TensorFlow Lite. En la figura 16 se representa la arquitectura de estos modelos.

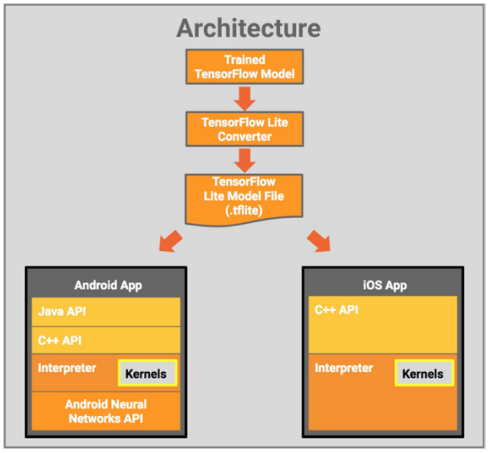


Figura 16. Arquitectura de la TensorFlow Lite para aplicaciones Android y para iOS[17]

A continuación se definen los modelos previamente entrenados compatibles con TensorFlow Lite con mejores resultados:

- Inception V3. Modelos destinados para la detección de objetos presentes en una imagen.

17 https://medium.com/ai-learners/tensorflow-dev-summit-resumen-3e77862f5416

- MobileNets. Modelos destinados para la clasificación, detección, incrustación y segmentación. Son modelos más pequeños y de menor precisión. En este proyecto se utiliza un modelo de este tipo.

2.6.2 TensorFlow Hub

La necesidad de este proyecto de utilizar un modelo preestablecido debido a la falta de recursos informáticos y grandes *datasheets,* hace imprescindible el uso de TensorFlow Hub. TensorFlow Hub es una herramienta que permite utilizar modelos previamente entrenados. Esta herramienta consta de una biblioteca destinada a la publicación, descubrimiento y consumo de partes reutilizables de modelos de aprendizaje automático.[18]

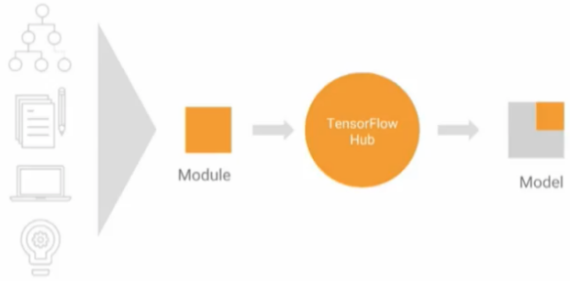


Figura 17. Funcionamiento de TensorFlow Hub [18]

En TensorFlow Hub, los módulos son partes independiente de un gráfico generado por TensorFlow que se pueden reutilizar en diferentes tareas específicas. Dentro de esto proyecto se hará uso de uno de ellos.

2.7 Marco regulador

2.7.1 Normativa y legislación

En esta sección y antes de comenzar con el desarrollo de las aplicaciones, se realiza un análisis de la legislación vigente con la que se puede hacer frente a los riesgos que supone el IoT para los usuarios. El objetivo principal es atajar estos riesgos desde el punto de vista de la protección de datos y de la privacidad analizando las soluciones que ofrece el marco legal relacionado con las amenazas que surgen del Internet de las cosas.

18 https://medium.com/ai-learners/tensorflow-dev-summit-resumen-3e77862f5416

La gran cantidad de datos y su flujo es una de las ventajas del IoT. Cada persona y su actividad se convierten en una fuente inagotable y constante de datos personales susceptibles. Debido al uso de estos datos personales para análisis y envíos entre dispositivos, es necesario regular la repercusión que el Internet de las cosas tiene y puede llegar a tener.

- Normativa Nacional:

La regulación del IoT consiste en determinar decisiones tanto sobre los objetos y sus datos que se conectan como sobre las redes y su seguridad. A continuación citaremos algunas leyes y normativas relacionadas:

La constitución Española de 1978 establece en el artículo 18 el derecho a la intimidad de las personas en la sección 1ª :

"Se garantiza el derecho al honor, a la intimidad personal y familiar y a la propia imagen."

Por otro lado en la sección 4ª :

"La Ley limitará el uso de la Informática para garantizar el honor y la intimidad personal y familiar de los ciudadanos y el pleno ejercicio de sus derechos."

En La Ley Orgánica Española 15/1999 [19], del 13 de diciembre, de Protección de Datos de Carácter Personal (LOPD) se regula el tratamiento de datos y ficheros de carácter persona, los derechos de los ciudadanos sobre dichos y las obligaciones de aquellos que los tratan. Algunos de sus artículos relevantes son lo siguientes:

Artículo 6.1: Consentimiento del afectado

"El tratamiento de los datos de carácter personal requerirá el consentimiento inequívoco del afectado, salvo que la ley disponga otra cosa"

Artículo 12: Acceso a los datos por cuenta de terceros

"*No se considerará comunicación de datos el acceso de un tercero a los datos cuando dicho acceso sea necesario para la prestación de un servicio al responsable del tratamiento."*

- Normativa Europea:

En cuanto a nivel Europeo, la Carta de Derechos Fundamentales de la Unión Europea recoge en su artículo 8, *"Protección de datos de carácter personal":*

> *1. Toda persona tiene derecho a la protección de los datos de carácter personal que la conciernan.*
> *2. Estos datos se tratarán de modo leal, para fines concretos y sobre la base del consentimiento de la persona afectada o en virtud de otro fundamento legítimo previsto por la ley. Toda persona tiene derecho a acceder a los datos recogidos que la conciernan y a su rectificación.*
> *3. El respeto de estas normas quedará sujeto al control de una autoridad independiente.*

El nuevo Reglamento Europeo de Protección de Datos, Reglamento (UE) 2016/679 [20], el cual entró en su plena aplicación el 25 de mayo de 2018. Relativo a la protección de las personas físicas en lo que respecta al tratamiento de datos personales y a la libre circulación de estos datos y por el que se deroga la Directiva 95/46/CE. Los puntos principales de este nuevo Reglamento se centran en el análisis de impacto de privacidad, en el deber de información, el consentimiento, la transparencia, la seguridad y en garantizar los derechos de los ciudadanos en relación con la protección de su privacidad.

Dentro del artículo 47 de La Directiva 2014/53/UE [21] relativa a la armonización de las legislaciones de los Estados miembros sobre la comercialización de equipos radioeléctricos, y por la que se deroga la Directiva 1999/5/CE, establece lo siguiente:

> *a) garantizar el establecimiento de un sistema coherente a nivel de la Unión para todos los equipos radioeléctricos;*
> *b) permitir la convergencia del sector de las telecomunicaciones, el audiovisual y el de las tecnologías de la información; L 153/92 ES Diario Oficial de la Unión Europea 22.5.2014*
> *c) permitir que las medidas regulatorias se armonicen a nivel internacional;*
> *d) lograr un alto nivel de protección de los consumidores;*
> *e) garantizar que los equipos radioeléctricos portátiles interactúen con los accesorios, en particular con los dispositivos de carga comunes;*
> *f) permitir, cuando los aparatos radioeléctricos estén dotados de una pantalla integral, que se presente en ella la información requerida.*

La Directiva 2013/40/UE [22], relativa a los ataques contra los sistemas de información, aborda el tema de la cibercriminalidad. En ella se aproxima las normas del Derecho penal de los Estados Miembros para tratar los de ataques contra los sistemas de información, mediante el establecimiento de normas mínimas, sanciones aplicables y mejoras en la cooperación entre las autoridades. El amplio tema de la ciberseguridad se trata en la reciente Directiva de Seguridad en Redes y Sistemas de Información .

Por último nombrar la Iniciativa emblemática de Europa 2020: *Unión por la innovación* [23]. Esta iniciativa mantiene un enfoque estratégico sobre la innovación que pretende mejorar las condiciones y el acceso a la financiación del I+D.

Debido al rápido desarrollo y crecimiento de esta tecnología, el camino que tiene que recorrer su marco regulador cambia continuamente. Este deberá desembocar en una legislación flexible hacia esta tecnología pero siempre teniendo en cuenta la protección de los usuarios. A pesar de que regular la titularidad de los datos es una tarea muy compleja la Unión Europea deberá seguir lanzando iniciativas como, La Iniciativa de Comunidades Conectadas [24], para impulsar y promover el desarrollo del IoT.

2.7.2 Impacto del marco regulador en el proyecto

En esta segunda sección dentro del marco regulador se analiza el impacto que tiene esta normativa tanto nacional como Europea en el proyecto. En primer lugar cabe recordar que las aplicaciones realizadas están basadas en un hardware de desarrollo. Esto quiere decir que, en principio, estos desarrollos no tienen porque sufrir ningún riesgo relacionado con la protección de datos y la privacidad.

Uno de los objetivos de cada una de estas aplicaciones es su posterior puesta en producción, por lo que a continuación se destacan los principales puntos que se han considerado previamente al desarrollo de estas, previniendo su posterior implementación en un entorno de producción.

- Para la elección de una plataforma de almacenamiento y acceso de datos en tiempo real para la primera aplicación se ha tenido en cuenta tanto la Ley Orgánica Española 15/1999 como el nuevo Reglamento Europeo de Protección de Datos, Reglamento (UE) 2016/679. Finalmente se eligió Firebase al ser una plataforma de la misma compañía asegurando seguridad y protección de datos.

- Otra de las razones por la que se ha elegido Firebase ha surgido debido al artículo 12 de La Ley Orgánica Española 15/1999 en relación al acceso a los datos por cuenta de terceros. La razón es la futura implementación en la aplicación del servicio de Autentificación que ofrece Firebase.

- A la hora de elección del modelo pre-entrenado para el diseño del sistema de clasificación utilizado en la aplicación de Android Things con TensorFlow se ha tenido en cuenta La constitución Española eligiendo un modelo oficial para asegurar la intimidad de las personas y el derecho al honor evitando poder hacer detección de personas o objetos personales con el uso de otros modelos no oficiales.

- A la hora de la implementación y la elección tanto de la Raspberry como de la utilización de un Smartphone y de una pantalla HDMI se ha tenido en cuenta el artículo 47 de La Directiva 2014/53/UE relativa a la armonización de las legislaciones de los Estados miembros sobre la comercialización de equipos radioeléctricos.

2.8 Conclusiones sobre el estado del arte

Para concluir en análisis del estado del arte y con ello finalizar la mayor parte del trabajo de investigación se destacan a continuación los puntos y las conclusiones mas relevantes de los distintos apartados anteriores junto con su relación con los desarrollo posteriores.

- El sistema operativo Android junto con iOS abarcan el 99% del mercado actual. Mientras que el mismo Android tiene el 85,9% (2017). Este es uno de lo motivos por los cuales se ha llevado a cabo este proyecto con lenguaje Android.

- La última versión estable de Android, 8.0 Oreo, va ganando fuerza gracias a la ampliación de servicios como la API para redes neuronales. Gracias a esta versión se ha desarrollado el sistema de clasificación utilizado en una de las aplicaciones desarrolladas en este trabajo.

- Android Things amplia la arquitectura de Android Mobile incorporando APIs proporcionadas por la biblioteca de IoT. Gracias a esto se han utilizado distintas APIs para integrar otras tecnologías en los desarrollos.

- Las actualizaciones de Android Things son gestionadas por Google y se realizan a través de una consola con el fin de que sean seguras. Esto ha evitado retrasos a causa de estas actualizaciones a lo largo del proyecto.

Android Things es considerado como una oportunidad de crecimiento para la su empresa, aún así tiene competidores fuertes como: Windows IoT de Microsoft y Greengrass de Amazon, por lo que para liderar el mercado con esta tecnología debe darse prisa y combinar este software con otros productos y al mismo tiempo conectarlo a la nube.

Capítulo 3

Configuración del entorno de desarrollo

3.1 Introducción

A lo largo de este capítulo se describirán los pasos a seguir para obtener el entorno de desarrollo necesario para poder realizar las implementaciones de cada una de las aplicaciones.

Como se ha descrito en el capítulo 2, el entorno que se ha utilizado para la implementación de este proyecto es un entorno de desarrollo. Por lo tanto, este cuenta con una configuración y un hardware específico para este tipo de entornos.

A continuación se destacan los elementos utilizados para iniciar la configuración del entorno de desarrollo utilizado en el proyecto:

- Android Studio. Versión 3.1
- Raspberry Pi 3 Starter Kit
- MacBook Pro. Versión 10.11.6 OS X El Capitan
- Acceso a Wifi
- Pantalla HDMI

Para una correcta configuración de este entorno de desarrollo se han llevado a cabo una serie de pasos, descritos en la página oficial [25], que se describen a continuación:

- Descarga de la versión del sistema operativo Android Things mas actual. Este paso se encuentra descrito en el anexo II.
- Flashing the image. Instalar la imagen del sistema operativo de Android Things descargado en la Raspberry Pi.
- Descarga e instalación de los comandos ADB.
- Conectar Hardware.
- Configuración del servicio Wifi.

Las constantes actualizaciones del sistema operativo utilizado durante el proyecto ha provocado la utilización de varias versiones de Android Things en el desarrollo de las aplicaciones. La versión de la imagen de Android Things mas reciente que se ha utilizado y con la que se ha finalizado este proyecto es 1.0.3.

Antes de comenzar a detallar los pasos de la configuración del entorno de desarrollo se destaca la necesidad de utilizar, en algunos de estos pasos, un software específico para MacBook.

3.2 Flashing the image

Antes de comenzar a flashear la imagen en el sistema de Hardware compatible, en nuestro caso la Raspberry Pi, se debe disponer de lo siguiente: cable micro-USB, cable Ethernet, lector de tarjetas MicroSD, tarjeta MicroSD de 8GB o superior, cable HDMI (opcional) y pantalla habilitada para HDMI (opcional).

El primer paso es descargar el software facilitado por Android Things Console para el flasheo de imágenes[19], en este proyecto se utilizará la versión 1.0.19 de este software. Una vez descargado se descomprime y se inicia la configuración con el siguiente comando en el terminal (en Mac) obteniendo los posteriores resultados:

```
MacBook-Pro-de-David-3: $ sudo ~/Downloads/androis-things-setup-
utility/Android-things-setup-utility-macos
```

Código 1. Comando para iniciar *Android Things Setup Utility* por consola

```
Android Things Setup Utility (version 1.0.19)
==============================
This tool will help you install Android Things on your board and set up Wi-Fi.
```

19 https://partner.android.com/things/console/u/0/#/tools

```
What do you want to do?
1 - Install Android Things and optionally set up Wi-Fi
2   - Set up Wi-Fi on an existing Android Things device
1
What hardware are you using?
1 - Raspberry Pi 3
2 - NXP Pico i.MX7D
3 - NXP Pico i.MX6UL
1
You chose Raspberry Pi 3.

Setting up required tools...
Fetching additional configuration...
Downloading platform tools...
File already downloaded.
Unzipping platform tools...
Finished setting up required tools.

Raspberry Pi 3
Do you want to use the default image or a custom image?
1 - Default image: Used for development purposes. No access to the Android
Things Console features such as metrics, crash reports, and OTA updates.
2 - Custom image: Upload your custom image for full device development and
management with all Android Things Console features.
1
Downloading Android Things image...
File already downloaded.
Unzipping image...

Downloading Etcher-cli, a tool to flash your SD card...
File already downloaded.
Unzipping Etcher-cli...

Plug the SD card into your computer. Press [Enter] when ready

Running Etcher-cli...
? Select drive /dev/disk1 (15.9 GB) - SD Card Reader
? This will erase the selected drive. Are you sure? Yes
Flashing [========================] 100% eta 0s
Validating [========================] 100% eta 0s
iot_rpi3.img was successfully written to SD Card Reader (/dev/disk1)
Checksum: e5b2d28d

If you have successfully installed Android Things on your SD card, you can now
put the SD card into the Raspberry Pi and power it up. Otherwise you can abort
and run the tool again.
```

Código 2. Configuración ejemplo de una imagen de *Android Things*

Al comienzo de este proyecto aún no existía este software por lo que se utilizó otro distinto para flashear la imagen, denominado *SD Card Formatter*.

3.3 Introducción comandos ADB

Una de las ventajas que tiene Android como lenguaje es su versatilidad. A veces, cuando tocamos los archivos internos del sistema como flashear una ROM o conseguir

permisos de administrador, no se encuentran programas adecuados a nuestro terminal por lo que se hace uso de otros métodos.

ADB (*Android Debug Bridge*) es una herramienta que se utiliza para interactuar con nuestro hardware compatible de una forma más avanzada y completa. A continuación se definen los comandos de esta herramienta utilizados en esta aplicación a través de la consola.

- `adb devices`. Muestra un listado de los dispositivos conectados con su respectivo número de serie y estado.
- `adb install`. Instala aplicaciones en formato apk en el dispositivo.
- `adb connect`. Conecta a la ip de tu dispositivo.

3.4 Conectar Hardware

Para poder comenzar a compilar distintos desarrollos en la Raspberry se necesita realizar su conexión.

En primer lugar se inserta la tarjeta microSD en la ranura microSD en la parte inferior de la Raspberry Pi. Posteriormente se conecta la Raspberry a la alimentación y a la red local donde se le asignará una dirección IP. Al conectarla a una pantalla se obtiene el siguiente resultado:

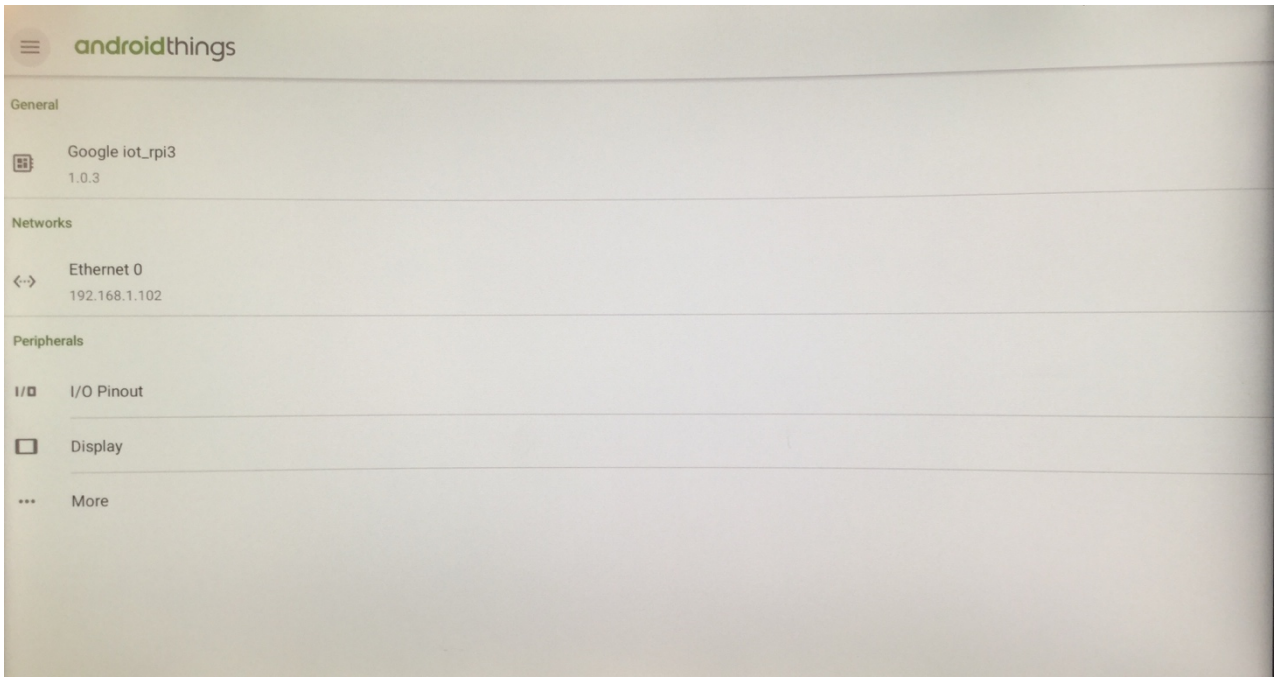


Figura 18. Pagina inicio de la Raspberry con el sistema operativo de Android Things

En la pantalla de inicio de Android Things se encuentran tanto varias características del sistema operativo como distintos parámetros de configuración. Algunos de estos datos que ofrece el software de Android Things son:

- Una pestaña con los datos más relevantes del sistema como su versión y su API (Figura 17). Dentro de esta pestaña también ofrece la posibilidad de actualizar la versión del sistema operativo a través de servidores OTA.
- Pestaña que recoge la información de la red a la que está conectada, ofreciendo la posibilidad de configurar la wifi.
- Finalmente ofrece una sección dedicada a los periféricos disponibles en el hardware utilizado. Esta incluye la lista de pines disponibles, visión a tiempo real de la captura del módulo de la cámara (solo si se dispone de una), sistema de comprobación del funcionamiento de la Rainbow HAT y la posibilidad de comprobar el funcionamiento del display.

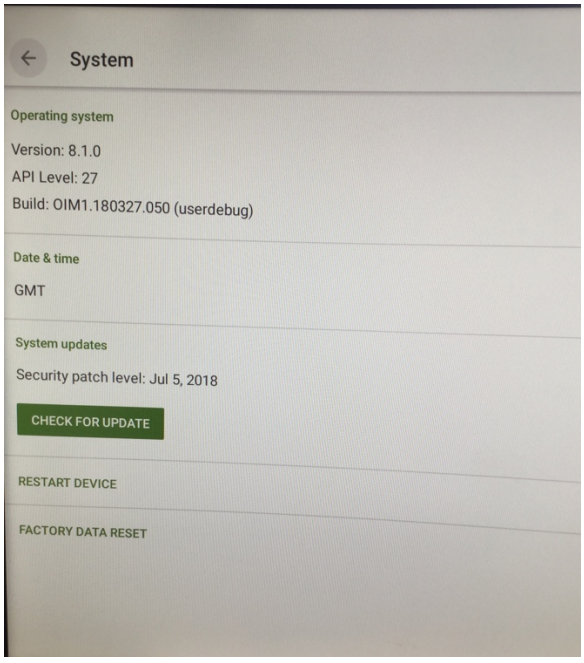


Figura 19. Pestaña para realizar las actualizaciones del software

Para su conexión se descarga el paquete de los comandos adb para Mac, *SDK Platform-Tools for Mac* [20] y se instalará de la siguiente manera:

```
MacBook-Pro-de-David-3: $ export
PATH=${PATH}:/Users/DavidGonzalezRsmos/Desktop/TFG/platform-tools
```

Código 3. Comando de descarga de *Platform-Tools for Mac* a través de consola

20 https://developer.android.com/studio/releases/platform-tools.html

Para finalizar se realizará la conexión a la dirección IP asignada a la Raspberry y estará lista para su utilización. El siguiente ejemplo ilustrará los comandos necesarios:

```
MacBook-Pro-de-David-3: $ adb devices
List of devices attached

MacBook-Pro-de-David-3: $ adb connect 192.168.X.XXX
Conected to 192.168.X.XXX
```

Código 4. Conexión a la Raspberry a través de consola

3.5 Wi-Fi con la Raspberry

Existen varias maneras de conectar la Raspberry Pi a una Wi-Fi:

- A través de *Android Things Setup Utility,* donde se obtiene los siguientes resultados por consola:

```
Would you like to set up Wi-Fi on this device? (y/n) y
Please plug your Raspberry Pi to your router with an Ethernet cable, then press [Enter].

Attempting to connect to your Raspberry Pi at Android.local...


Connected to device through Ethernet.
Enter the Wi-Fi network name:
Enter the Wi-Fi network password (leave empty if no password):
Connecting to Wi-Fi network XXXXX
```

Código 5. Conectar la Raspberry Pi a una Wi-Fi a través de *Android Things Setup Utility*

- A través de la aplicación de inicio después de conectar la pantalla.
- A través de los comando *adb*

Capítulo 4

Implementación de dos proyectos sobre Android Things

4.1 Visión General

A lo largo del capítulo 2 se ha tratado de explicar la situación actual del Internet de las cosas, realizando un recorrido desde sus inicios hasta su actualidad centrándose sobre todo en el sistema operativo de Android. También se ha realizado una introducción a Android Things, el nuevo sistema operativo para el Internet de las Cosas y su Hardware compatible.

El estado del arte es una fase fundamental de este proyecto debido a que, al trabajar con una tecnología relativamente nueva, el proceso de investigación y documentación es una parta imprescindible para el correcto desarrollo de las distintas aplicaciones.

Una vez afianzados estos conceptos generales, en este capítulo se detallan todos los aspectos importantes en relación a las dos aplicaciones. Como se ha mencionado anteriormente, la primera de ellas consiste en una aplicación sencilla para el control de distintos periféricos de la Raspberry Pi en tiempo real a través del servicio de *RealTime DataBase* de *Firebase* implementado en una aplicación móvil. En la segunda se diseña, usando las dependencias de Tensorflow y con ayuda del módulo de una cámara, un clasificador de imágenes a través de la exportación de un modelo ya entrenado y empaquetado. El desarrollo de ambas aplicaciones se ha realizado en el entorno de desarrollo oficial Android, Android Studio.

La progresiva introducción a Android Things que ofrecen estas dos aplicaciones y su facilidad de integración con otras tecnologías han sido los motivos por el cuales se han decidido realizar estas dos aplicaciones por encima de otras.

Para realizar una correcta y ordenada exposición de ambas aplicaciones, este capítulo se divide en dos puntos dedicados a cada una de ellas.

En cada uno de estos puntos se definen tanto algunos aspectos teóricos necesarios para las implementaciones, como las distintas dependencias y módulos utilizados en cada uno de los desarrollos.

4.2 Android Things con Firebase

4.2.1 Introducción al proyecto

Este primer proyecto tiene como objetivo principal la introducción en el diseño de aplicaciones para Android Things. Por otro lado y a parte de los distintos objetivos principales del trabajo, el diseño y desarrollo de esta primera aplicación tiene los siguientes objetivos particulares:

- Adquirir los conocimientos necesarios para el control y el manejo de los distintos pines GPIO de la Raspberry Pi.
- Aprender a integrar la biblioteca de Firebase en distintos proyectos Android, tanto en dispositivos móviles como en el Hardware compatible de Android Things.
- Adquirir los conocimientos necesarios para el uso del servicio *RealTime Database* proporcionada por Firebase.

Para el comienzo de su desarrollo se hace uso de un ejemplo proporcionado por Android a través de su página web[21]. En él se muestra cómo utilizar un botón de entrada *UserDriver*, escuchar los cambios de pines GPIO y generar eventos para cambiar el estado de un LED. A lo largo del desarrollo de esta aplicación se han realizado numerosas modificaciones y añadidos sobre este ejemplo hasta conseguir la funcionalidad final deseada. Las más importantes se destacan a continuación:

- Integración completa del proyecto con Firebase, añadiendo tanto las dependencias como los archivos necesarios.
- Desarrollo de la lógica necesaria para el control y el manejo de un led y del display.
- Desarrollo de una aplicación móvil para el control de varios periféricos de la Raspberry.

Para llevar a cabo el desarrollo de estas mejoras se debe tener en cuenta los siguientes requisitos fundamentales:

21 https://developer.android.com/samples/

- Raspberry Pi 3 Starter Kit (equipamiento de partida)
- Cuenta de Gmail
- Dispositivo móvil. En este proyecto con Versión Android 6.0.1
- Android Studio con versión mínima 2.2
- Proyecto en Firebase
- Conexión WiFi

En cuanto al diseño final de este primer proyecto consiste en una aplicación para la Raspberry integrada con Firebase en donde se define el uso de los pines GPIO correspondientes a un led y al display de la Raspberry. Esta integración permite hacer uso de su servicio de almacenamiento de información en su base de datos y sincronizar la aplicación con dicha base de datos en tiempo real.

Otra aplicación para un dispositivo móvil, también integrada con la misma base de datos de Firebase, en la cual se diseña una interfaz que ofrece varios controles sobre los pines definidos en la aplicación de la Raspberry.

El conjunto de estas dos aplicaciones ofrece un sistema de control remoto que permite controlar a través de un dispositivo móvil varios pines de la Raspberry en tiempo real. A lo largo de esta sección se expone tanto la estructura como las distintas partes añadidas más importantes de este sistema de control remoto basado en Firebase. En la siguiente imagen se representa el esquema de la aplicación.

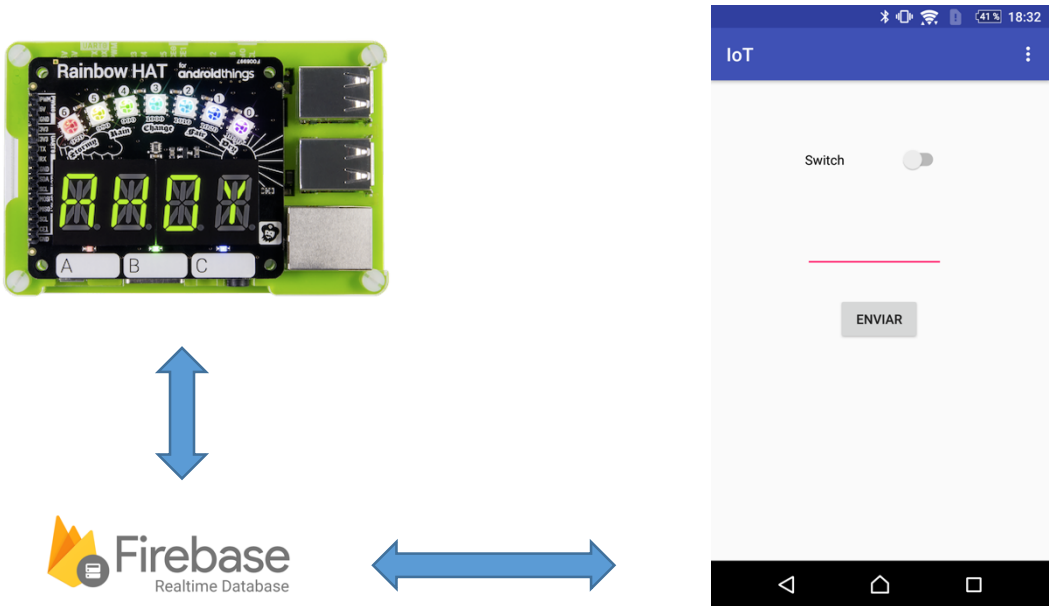


Figura 20. Arquitectura de la primera aplicación

4.2.2 Configuración de la conexión con los periféricos

Android Things tiene una ventaja única a la hora de conectarse a componentes electrónicos externos. Esta conexión la realiza fácilmente a través de su Peripheral API y su soporte integrado para dispositivos. Esta API, denominada *PeripheralManagerService* permite encontrar todos los puertos disponibles y abrir o cerrar cualquiera de ellos para cualquier aplicación. En general se usa para la comunicación con sensores y actuadores utilizando protocolos e interfaces estándar. Algunos de estos son los siguientes:[26]

- GPIO. Los pines de Entrada/Salida GPIO proporcionan una interfaz programable para leer el estado de un dispositivo de entrada binaria. Se utiliza con sensores simples como un led o el interruptor de un botón y su uso solo requiere un pin y valores booleanos para estados alto y bajo.

- PWM. La modulación de ancho de pulso es comúnmente usado para aplicar una señal de control proporcional a un dispositivo externo usando un pin de salida digital. Se utiliza para servomotores, pantallas LCD para el ajuste de brillo a través del valor promedio de una señal PWM.

Como se ha mencionado lo que busca este proyecto es el control en tiempo real a través de un dispositivo móvil varios pines de Entrada/Salida GPIO como un led y el display. Ahora se muestra la forma de definir el servicio de *PeripheralManagerService* y de como obtener la referencia GPIO de un pin concreto.

```
public class ControlActivity extends Activity {

    private static final String GPIO_NAME = ...;
    private Gpio mGpio;
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
 PeripheralManagerService manager = new PeripheralManagerService();
        try {

            mGpio = manager.openGpio(GPIO_NAME);
        } catch (IOException e) {
             Log.w(TAG, "Unable to access GPIO", e);
        }
    }
    protected void onDestroy() {
        super.onDestroy();

        if (mGpio != null) {
            try {
                mGpio.close();
                mGpio = null;
            } catch (IOException e) {
                Log.w(TAG, "Unable to close GPIO", e);
            }
        }
    }
}
```

Código 6. Obtener referencia de un pin GPIO

Una vez realizada la conexión y las acciones con el puerto GPIO se debe cerrar la conexión para liberar recursos. De esta manera, siempre que se abra una conexión con cualquier puerto se debe definir una función *onDestroy()* para cerrarla en el momento deseado.

Para poder controlar a través de código el estado de los puertos GPIO se debe configurar dicho puerto como salida usando el método *setDirection (),* el cuál puede tomar los siguientes valores:

- DIRECTION_OUT_INITIALLY_HIGH : establece el valor de salida inicial en alta tensión (encendido).
- DIRECTION_OUT_INITIALLY_LOW : establece el valor de salida inicial a bajo voltaje (apagado).

Para poder escribir en el dispositivo, puedes usar *setValue(boolean)* en el objeto Gpio para establecer el estado de su componente.

```
mGpio.setDirection(Gpio.DIRECTION_OUT_INITIALLY_LOW);
mGpio.setValue(true);
```

Código 7. Ejemplo de configurar la dirección y establecer el estado de un pin GPIO

En este primer proyecto se hace uso del kit completo *Raspberry Pi 3 Starter Kit,* tanto de las Raspberry como de la Rainbow HAT. Gracias a la Rainbow HAT la implementación del entorno de desarrollo relacionado con el Hardware se simplifica mucho. Para realizar y dejar listas las conexiones solo es necesario conectar la Rainbow HAT a la Raspberry.

En cuanto al control remoto de los distintos pines GPIO se realiza una integración de Android Things con Firebase. Como se ha visto, Android Things es una rama del sistema operativo Android, por lo tanto, Android Things es compatible con Firebase de forma inmediata.

4.2.3 Estructura y módulos del proyecto

Para una correcta exposición de los distintos pasos seguidos en el desarrollo del proyecto se divide el mismo en tres módulos o fases.

- El primer módulo consiste en la base de datos de Firebase en dónde se almacenarán los distintos estados. En esta primera fase se describe la creación de un proyecto en Firebase y como generar los archivos necesarios.

- El segundo módulo lo forma la aplicación de la Raspberry y la integración de Android Things con Firebase
- El tercer módulo consiste en la aplicación móvil integrada con Firebase

4.2.3.1 Creación de un proyecto en Firebase

Antes de agregar Firebase a las aplicaciones es necesario la creación de un proyecto y un archivo de configuración de Firebase para cada aplicación. El primer paso para la creación de un proyecto en Firebase es acceder con una cuenta de Gmail a su consola.

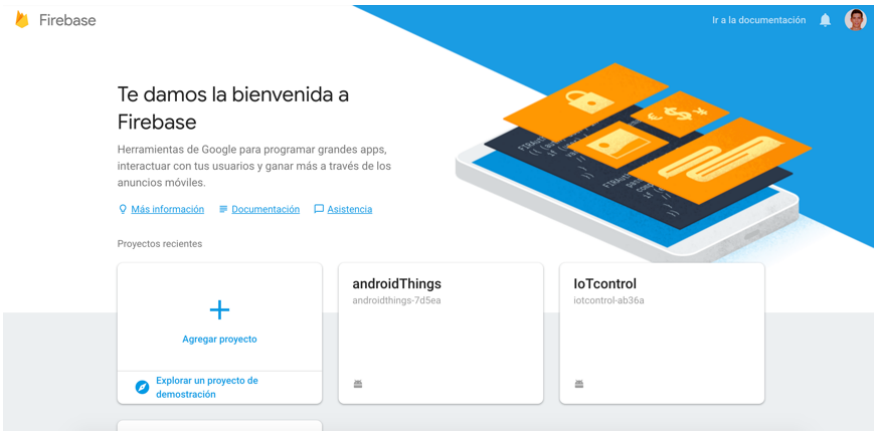


Figura 21. Consola Firebase

A continuación se accede al agregador de proyectos donde se debe introducir el nombre del proyecto. Se asigna un ID único al proyecto automáticamente, que se usa en características de Firebase visibles de manera pública.

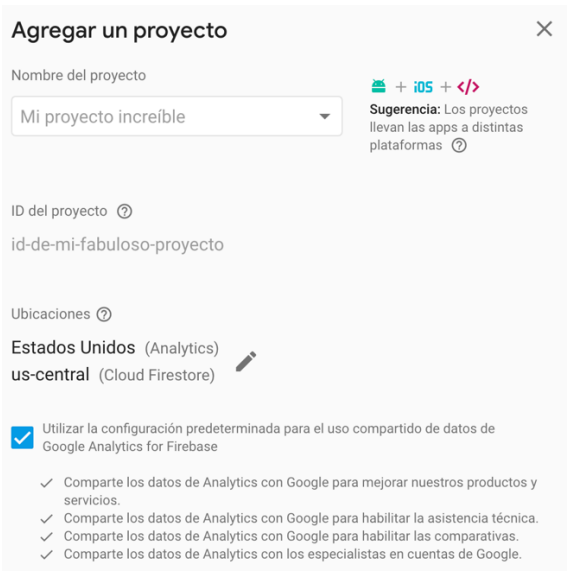


Figura 22. Interfaz para agregar un proyecto a Firebase

Una vez creado el proyecto se procede a agregar una aplicación Android. Dentro del proyecto creado se accede a *Agrega Firebase a tu app para Android* y se siguen los pasos de configuración. En uno de ellos se introduce el nombre del paquete de la app donde quieres integrar Firebase. A lo largo de este proceso se genera un archivo llamado *google-services.json*, el cuál debe ser añadido a nuestra aplicación.

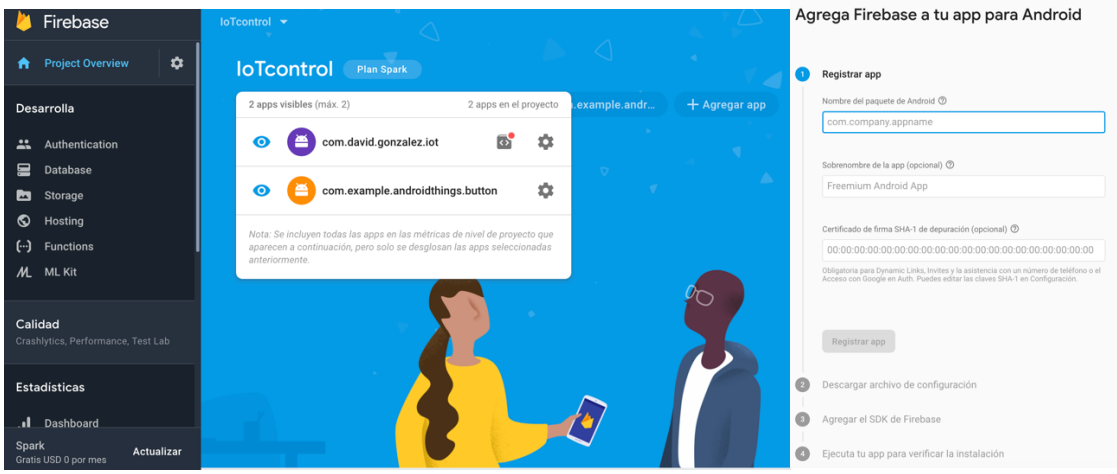


Figura 23. Interfaz para agregar Firebase a tu aplicación

Como se representa en la imagen anterior, al proyecto de Firebase utilizado en este primer proyecto con Android Things, llamado *IoTcontrol,* se le añaden dos aplicaciones.

Una de ellas se corresponde con la aplicación de la Raspberry por la cual se accede al servicio de *RealTime DataBase* de Firebase para modificar y escuchar el estado almacenado de los distintos pines GPIO. La otra aplicación es la correspondiente a la app del dispositivo móvil, la cual también necesita acceso al servicio de *RealTime DataBase* para realizar el control sobre los distintos periféricos.

En la siguiente imagen se muestra la configuración de la base de datos utilizada para almacenar el estados de los pines GPIO que queremos controlar de forma remota.

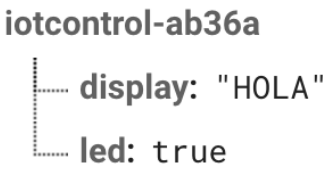


Figura 24. Configuración Database del proyecto de Firebase

Finalmente, para que las distintas aplicaciones asociadas a este proyecto de Firebase pueden acceder a los distintos valores almacenados en la base de datos es necesario modificar las reglas de seguridad como se muestra a continuación dónde se puede controlar los permisos de lectura y de escritura de la base de datos.

```
{
  "rules": {
    ".read": "true",
    ".write": "true"
  }
}
```

Figura 25. Reglas del servicio Database de Firebase

4.2.3.2 Aplicación para la Raspberry

Después de realizar la configuración necesaria para la creación de un proyecto en Firebase, se realiza el desarrollo de la aplicación de Android Things para la Raspberry.

Como se ha mencionado anteriormente, esta aplicación se basa en un ejemplo proporcionado por Android en el cual se implementa la conexión con el pin GPIO de un Led (correspondiente en la Rainbow HAT con "BCM6") y con un InputDriver (botón correspondiente en la Rainbow HAT con "BCM21"). Este ejemplo consiste en el control local del estado de un led al presionar un botón, ambos periféricos de la Raspberry. La lógica en la que se basa este ejemplo está descrita en la sección 4.2.2 en la parte del control de los pines GPIO.

En esta sección se describe las modificaciones realizadas desde el comienzo del desarrollo hasta la aplicación final. Para la realización de esta modificaciones se utilizan diferentes dependencias definidas en la página *github* oficial de Android Things[22]. La primera modificación que se realiza es la declaración del periférico del display, tipo *AlphanumericDisplay*, importando el paquete necesario y declarando su función onDestroy() correspondiente.

```
Import com.google.android.things.contrib.driver.ht16k33.Alphanume
ricDisplay;
[…]
private AlphanumericDisplay mSegmentDisplay;
[…]
try {
      mSegmentDisplay = new AlphanumericDisplay(I2C_BUS);
      mSegmentDisplay.setBrightness(1.0f)
      mSegmentDisplay.setEnabled(true);
      mSegmentDisplay.clear();
[…]
```

Código 8. Declaración y definición del display

22 https://github.com/androidthings/drivers-samples

El siguiente paso es la integración de Android Things con Firebase. Esta aplicación debe escuchar los cambios de los estados definidos en la base de datos de Firebase y reaccionar al control de estos. Para realizar esta integración se añade al proyecto las siguientes dependencias:

- Al archivo *build.gradle* del proyecto se le añade el las dependencias de *google-services* y el repositorio *Maven* de Google.

```
dependencies {
        // ...
        classpath 'com.google.gms:google-services:4.0.1'    }

repositories {
        maven {
            url 'https://maven.google.com/'
            name 'Google'
        }
```

Código 9. Código añadido a *build.gradle*

- En el archivo Gradle del módulo se le añade las siguientes líneas de código necesarias para utilizar el servicio de *RealTime database* de Firebase. Por otro lado también se añaden las dependencias para el uso del display.

```
dependencies {
  // ...
  compile 'com.google.firebase:firebase-database:16.0.1'
  compile 'com.google.android.things.contrib:driver-ht16k33:0.4'
}
apply plugin: 'com.google.gms.google-services'
```

Código 10. Código añadido al archivo Gradle del módulo

Antes de comenzar a escribir o leer de la base de datos, se recupera la instancia de nuestra base de datos a través de *getInstance()*. Con esta instancia se obtienen las referencias de cada una de las variables definidas en la base de datos y con ellas, podemos modificar sus valores.

```
FirebaseDatabase database = FirebaseDatabase.getInstance();
DatabaseReference referencia1 = database.getReference("led");
DatabaseReference referencia2 = database.getReference("display");
```

Código 11. Obtener referencias de la base de datos

Una vez obtenidas las referencias de cada uno de los periféricos y para escuchar los cambios de la aplicación en tiempo real, se agrega un *addValueEventListener* a cada una de las referencias y se sobrescribe el método *onDataChange* para escuchar cada vez que produzca un cambio en el estado de cada una de las variables de Firebase.

Por último, para poder lanzar eventos a la Rainbow HAT se utilizan funciones proporcionada por Android Things como *setLedValue(boolean)* para modificar el estado del Led o *display(String)* para escribir en el display. Se representan algunos resultados de la Raspberry.

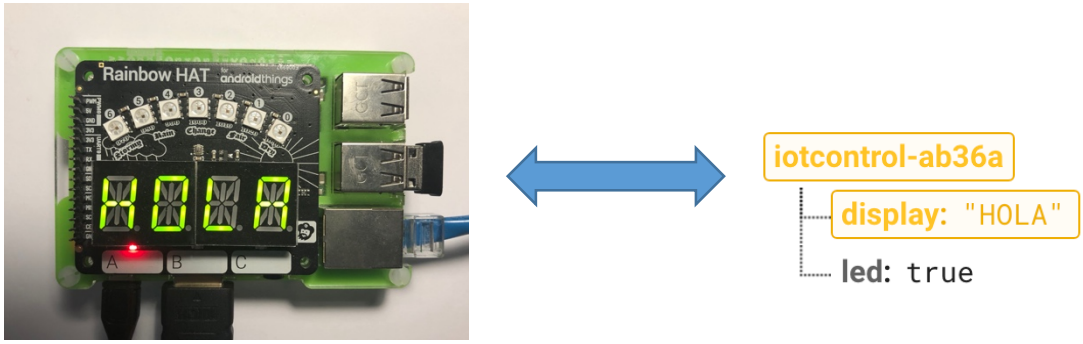


Figura 26. Raspberry con estado “on” en el Led y valor del display “HOLA”

4.2.3.4 Aplicación para dispositivo Android

En esta última fase del proyecto se describe el diseño de la solución técnica de la aplicación móvil. El objetivo de esta aplicación es el control de la estructura de datos definida en Firebase a través del diseño de varios elementos integrados con su servicio de *Realtime Database*. Para realizar esta integración se realiza una configuración similar a la de la aplicación de la Raspberry.

En cuanto a la arquitectura de esta app es bastante sencilla, aunque este tercer módulo de esta aplicación no implica ningún desarrollo de Android Things es una parte necesaria para el diseño del sistema de control remoto. Para su representación se realiza una división de la distintas partes que la conforman en distintas capas: la interfaz del usuario, la capa de software, la capa del sistema operativo y la capa de comunicación.

- La interfaz del usuario o la capa vista es la capa dónde se representa distintos actuadores con los que el usuarios puede comunicarse con el sistema. En esta aplicación, esta capa es muy simple ya que sólo recoge tres componentes con los que el usuario interactúa, esto permite una rápida comprensión por parte del usuario:

    - Un conmutador de dos estados en el que el usuario puede elegir entre ambas opciones arrastrando este elemento hacia adelante o hace atrás.

Ofrece el control sobre el encendido y apagado del pin GPIO correspondiente al Led.
- Componente que actúa como campo de texto donde el usuario puede editar su contenido. Permite modificar el display de la Rainbow HAT.
- Botón cuya funcionalidad está asociada al envío del contenido del campo de texto.

- En la capa del software interno se recogen las funcionalidades de la configuración inicial de los distintos componentes de la interfaz de usuario, la declaración de escuchadores (*addValueEventListener*) para establecer cambios en la aplicación según los cambios realizados a través de la plataforma de Firebase y la declaración de un servicio de notificaciones cada vez que se detecte un cambio en el estado de algún elemento.

  En cuanto al servicio de notificaciones está compuesto por un *notificationManager* asociado a una referencia de la base de datos de Firebase.

- La capa del SO es la correspondiente con el sistema operativo Android, encargado de la comunicación entre el dispositivo móvil y la aplicación. También dispone de las bibliotecas necesarias para realizar el desarrollo de los distintos servicios.

- La capa de comunicación, Internet, es la encargada de establecer la comunicación entre el SO del dispositivo y el servicio la base de datos de Firebase. Esta permite leer y modificar la estructura de datos definida en el proyecto.

A continuación se muestra los distintos estados y la comunicación entre el dispositivo móvil y los elementos de la base de datos.

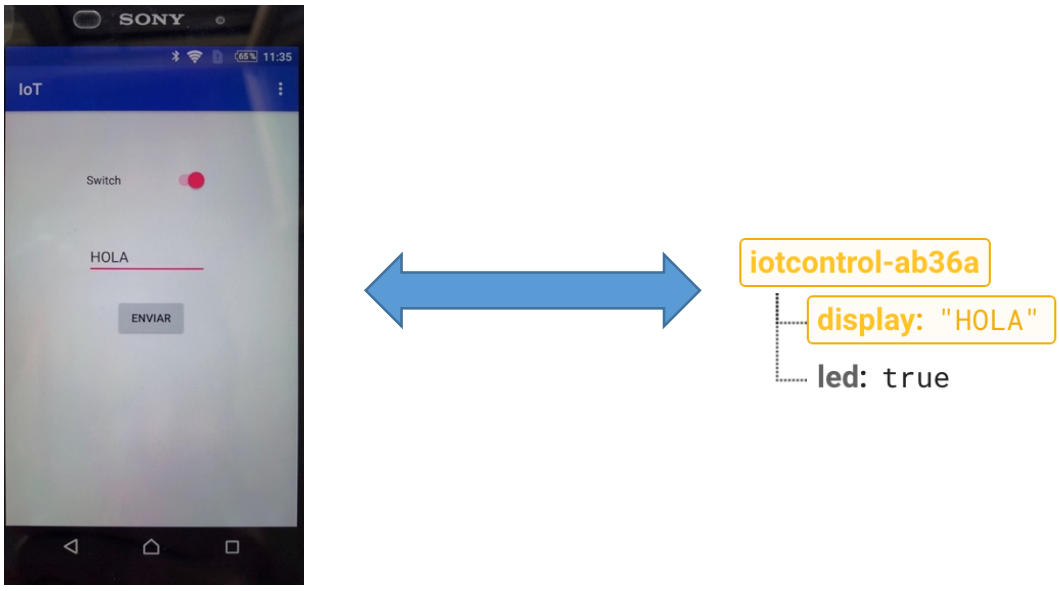


Figura 27. Estado “on” Led y valor del display “HOLA”

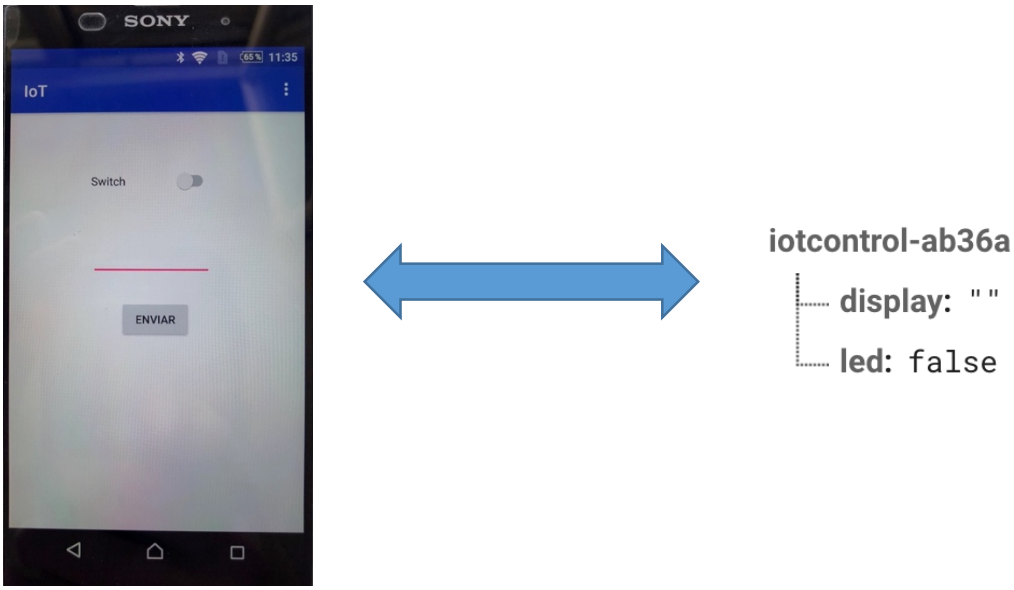


Figura 28. Estado "off" Led y display vacío

4.2.4 Compilación y resultados

Para obtener la aplicación final el último paso es la compilación de los distintos proyectos en sus respectivos dispositivos compatibles. La compilación de la aplicación móvil se ha realizado en un dispositivo móvil con versión de Android 6.0.1. Por otro lado, para poder compilar el proyecto final de la raspberry hay que asegurarse de tener conexión con ella y verificar que se puede acceder a dicho Hardware a través de los comandos adb.

En cuanto al proceso de evaluación de esta primera aplicación se ha llevado a cabo a lo largo de todo el desarrollo ya que se han realizado numerosas compilaciones por cada función implementada. Este proceso de evaluación también ha estado formado por las comprobaciones a través de la plataforma de Firebase.

Una vez realizados los procesos de desarrollo, compilación y evaluación ya se obtiene la aplicación final la cual recoge todos los objetivos descritos. Con la realización de este primer proyecto se ha obtenido el conocimiento sobre cómo integrar Android Things con Firebase para poder controlar en tiempo real la placa Raspberry Pi de forma remota. A partir de aquí se pueden desarrollar múltiples desarrollos para el control de cual periférico del que se disponga.

4.3 Android Things con TensorFlow

4.3.1 Introducción y requisitos del proyecto

Como se ha resumido en el capítulo anterior, Google permite integrar de una manera relativamente sencilla su sistema operativo del internet de las cosas, Android Things, con su plataforma de *Machine Learning* Tensorflow. A lo largo de esta sección se expone el desarrollo del segundo diseño con Android Things.

Esta segunda aplicación consiste en la implementar un motor de *Machine Learning* a un sistema de IoT, es decir, en la integración de TensorFlow en Android Things. Gracias a este segundo desarrollo se consigue evaluar el potencial de la combinación de ambas tecnologías en una solución final. Esta solución final consiste en la ejecución de la inferencia de TensorFlow Lite para Android creando un dispositivo capaz de capturar imágenes a través de una cámara y clasificarlas localmente a través de un modelo previamente entrenado.

A parte de los distintos objetivos principales del proyecto, el diseño y desarrollo de esta segunda aplicación tiene los siguientes objetivos particulares:

- Aumentar los conocimientos del manejo de los distintos pines GPIO de la Raspberry Pi.
- Adquirir los conocimientos necesarios para el uso de la API de la cámara de Android.
- Aprender a integrar la biblioteca de TensorFlow en distintos proyectos Android.
- Introducción a la carga de datos en un modelo de TensorFlow.

En el comienzo de esta aplicación también se hace uso de un ejemplo proporcionado por Android a través de su página oficial[23]. Sobre este ejemplo se realizan modificaciones tanto en el manejo de los periféricos (display, cámara, etc) como en el formato de salida de los resultados. Para llevar a cabo su desarrollo se debe tener en cuenta los siguientes requisitos fundamentales:

- Raspberry Pi 3 Starter Kit (equipamiento de partida)
- *Camera Board V2* oficial para Raspberry Pi
- Adroid Studio con una versión 3.0 o mayor
- Pantalla HDMI (opcional)

23 https://developer.android.com/samples/

En cuanto al diseño de este segundo proyecto consiste en una aplicación para la Raspberry en donde se define tanto el uso de un pin GPIO correspondiente a un botón, el cual generara un *KeyEvent* posteriormente procesado para generar la acción de tomar la foto, como las distintas funciones destinadas a la inicialización y carga del modelo clasificador.

A lo largo de esta sección se expone tanto la estructura como las distintas partes y funciones más importantes de este sistema de clasificación de imágenes basado en TensorFlow. En la siguiente imagen se representa es esquema de la aplicación.

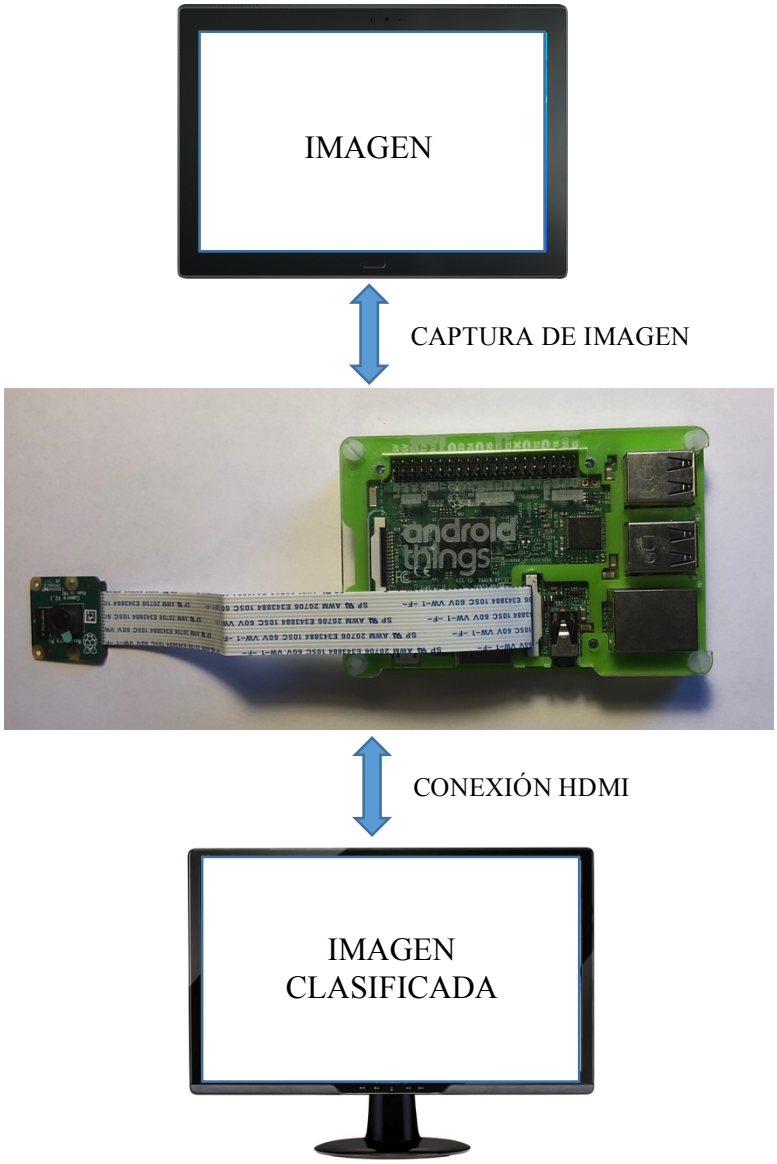


Figura 29. Flujo gráfico de la aplicación

4.3.2 Permisos y dependencias

Antes de comenzar con la exposición de este proyecto se resalta la necesidad de actualizar el sistema operativo Android a la versión 8.1 (API 27) para poder utilizar la API para inteligencia artificial que permite implementar sistemas de IA y *Machine Learning* en la aplicación. Para esto se lleva a cabo un nuevo flasheo de la imagen de la Raspberry previamente descargada.

Antes de comenzar con la descripción de los distintos módulos que forman la estructura inicial del proyecto se describen las partes que forman el manifiesto de la aplicación. En primer lugar se incluyen una serie de permisos que permiten leer desde almacenamiento externo, acceder a la cámara y manejar los controladores periféricos .

```
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.CAMERA"/>
<uses-permission
android:name="com.google.android.things.permission.USE_PERIPHERAL_IO"/>
<uses-permission
android:name="com.google.android.things.permission.MANAGE_INPUT_DRIVERS" />
```

Código 12. Permisos del *AndoridManifest.xml*

Dentro del manifiesto también se define también la única activity que forma parte de este proyecto, llamada *ImageClassifierActivity*.

```
<application
    […]
    <activity android:name=".ImageClassifierActivity"
        android:screenOrientation="landscape"
        android:label="@string/app_name">
        <intent-filter>
            […]
        </intent-filter>
    </activity>
</application>
```

Código 13. Definición de la *Activity* en *AndoridManifest.xml*

Por otro lado, en los archivos *Gradle* tanto de la aplicación como el del proyecto se incluye una configuración y una serie de dependencias para ejecutar el proyecto en la versión 1.0 de Android Things y para agregar TensorFlow al mismo.

```
allprojects {
    repositories {
        jcenter()
    }
}
aaptOptions {
    noCompress "tflite"
}

dependencies {
    compileOnly 'com.google.android.things:androidthings:1.0'
    implementation 'com.google.android.things.contrib:driver-rainbowhat:1.0'
    implementation 'com.android.support:support-annotations:27.1.0'
    implementation 'org.tensorflow:tensorflow-lite:0.1.7'

}
```

Código 14. Dependencias en los archivos *Gradle*

4.3.3 Estructura y módulos del proyecto

La estructura del proyecto está formada por distintos módulos básicos a los cuales se añadirán otros desarrollados posteriores. A continuación se procede a la exposición de estos módulos junto con sus agregaciones y varias modificaciones clasificándolos de la siguiente manera.

- Actividad principal: *ImageClassidierActivity*
- Dos clases destinadas a la cámara: CameraHandler y ImagePreprocessor
- Dos clases destinadas a TensorFlow: *TensorFlowHelper* y *Recognition*

4.3.3.1 Actividad *ImageClassidierActivity*

Este única actividad del proyecto la cual recoge la mayor parte del código básico de la aplicación. Esta actividad es la encarga de llevar a cabo las llamadas a las funciones principales y de definir el flujo principal de la aplicación. Seguidamente se procede a resaltar las partes mas importantes de este flujo.

El flujo principal comienza con una función *onCreate* dónde se define un contenedor (*RelativeLayout*) compuesto por un *imageView*, donde se representa la imagen capturada y por un *TextView*, dónde se representa el *string* que muestra el resultado de la clasificación.

Figura 30. Diseño del *layout activity_camera.xml*

A continuación, dentro de esta misma función se llaman a las funciones de inicialización de la cámara, del botón y de carga del modelo de clasificación. Estas funciones han sido implementadas más adelante, dónde se describe más concretamente su funcionalidad.

```
protected void onCreate(final Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    getWindow().addFlags(WindowManager.LayoutParams.FLAG_KEEP_SCREEN_ON);

    setContentView(R.layout.activity_camera);



    mImage = findViewById(R.id.imageView);
    mResultText = findViewById(R.id.resultText);

    updateStatus(getString(R.string.initializing));
    initCamera();
    initClassifier();
    initButton();
    updateStatus(getString(R.string.help_message));
}
```

Código 15. Función *onCreate*

Continuando con el flujo, otra función importante es onKeyUp. Esta función es llamada continuamente y describe la lógica que sucede al capturar el evento generado al pulsar el botón, *KeyEvent.KEYCODE_ENTER*. Esta lógica consiste en una llamada a la función destinada a hacer la captura de la foto definida en la clase CameraHandler.

```
public boolean onKeyUp(int keyCode, KeyEvent event) {
    if (keyCode == KeyEvent.KEYCODE_ENTER) {
        if (mProcessing) {
            updateStatus("Still processing, please wait");
            return true;
        }
        updateStatus("Running photo recognition");
        mProcessing = true;
        loadPhoto();
        return true;
    }
    return super.onKeyUp(keyCode, event);
}
```

Código 16. Función *onKeyUp*

4.3.3.2 Clases *CameraHandler* y *ImagePreprocessor*

Dentro de estas dos clases se recoge toda la lógica necesaria tanto para el control y el manejo de la cámara como para el tratamiento de imágenes indispensable para su posterior análisis por parte del sistema de clasificación.

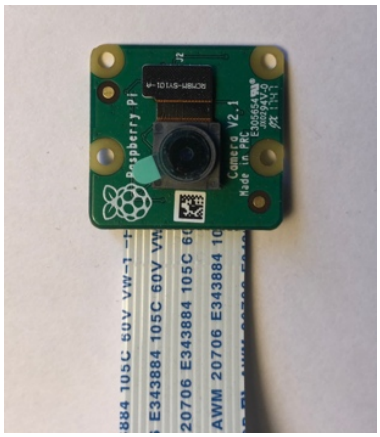


Figura 31. Camera Board V2 oficial para Raspberry Pi

- CameraHandler

Esta clase es destinada para la definición de las funciones correspondientes al inicio, al control de recursos y al manejo de estados de la cámara. El flujo principal que establece esta clase comienza con la importación de varios paquetes necesarios para la definición de distintas variables de control, como el máximo número de imágenes capturadas en un dispara o el control de inicialización y variables para el almacenamiento de imágenes.

```
[…]
import android.hardware.camera2.CameraCaptureSession;
import android.hardware.camera2.CameraCharacteristics;
import android.hardware.camera2.CameraDevice;

[…]

private static final int MAX_IMAGES = 1;
private CameraDevice mCameraDevice;
private CameraCaptureSession mCaptureSession;
private boolean initialized;
[…]
```

Código 17. La importación de varios paquetes y la definición de distintas variables de control

La siguiente función importante siguiendo con el flujo de esta clase es la encargada de inicializar el dispositivo de la cámara. Esta función se denomina *initializeCamera* y es el lugar dónde se captura la instancia de la cámara y se le asigna sus recursos. Su definición se encuentra en el anexo IV.

Como se ha mencionado anteriormente la función más importante definida dentro de esta clase es la encargada de capturar imágenes. Dentro de esta función se crea una *cameraCaptureSession* para capturar imágenes fijas.

```
public void takePicture() {
    if (mCameraDevice == null) {
        Log.w(TAG, "Cannot capture image. Camera not initialized.");
        return;
    }
    try {
        mCameraDevice.createCaptureSession(
                Collections.singletonList(mImageReader.getSurface()),
                mSessionCallback,
                null);
    } catch (CameraAccessException cae) {
        Log.e(TAG, "Cannot create camera capture session", cae);
    }
}
```

Código 18. Función *takePicture*

Dentro de esta clase también se encuentran definidas otro tipos de funciones destinadas al cierre del dispositivo de la cámara, a la recuperación de sus recursos y al control de los cambios de su estado. Estas últimas se denominan *Callback.*

- ImagenPreprocessor

Una vez capturada la imagen, esta segunda clase se encarga de su procesamiento y de extraer un mapa de bits en el formato apropiado para aplicar posteriormente el sistema de clasificación.

Para realizar esta función lo primero es la importación del paquete que permite el manejo de las funciones para realizar esta conversión de imágenes a mapa de bits.

```
import android.graphics.Bitmap;
import android.graphics.Bitmap.Config;
import android.graphics.BitmapFactory;
```

Código 19. Importaciones *Bitmap*

Gracias a esto se puede utilizar las siguientes funciones necesarias:

- `createBitmap(int width, int height, Bitmap.Config config)` La cual devuelve un mapa de bits mutable con el ancho y alto especificados.
- `compress(Bitmap.CompressFormat format, intquality, OutputStream stream).` Escribe una versión comprimida del mapa de bits en la salida especificada.
- `decodeStream(InputStream is).` Decodifica un flujo de entrada en un mapa de bits.

- Agregar cámara sobre el flujo principal

  Una vez expuestas las clases destinadas al control de la cámara y a la preparación de las imágenes, se implementan la función y los métodos necesarios para la utilización de estas en el flujo principal de la activity *ImageClassidierActivity.*

  En primer lugar se definen dos instancias de las clases anteriores para manejarlas dentro de estas funciones agregadas. Estas funciones son *initCamera, closeCamera y loadPhoto,* definidas en el anexo V.

```
private CameraHandler mCameraHandler;
private ImagePreprocessor mImagePreprocessor;
```

Código 20. Instancias de las clases destinadas al control de la cámara

4.3.3.3 Clases *Recognition y TensorFlowHelper*

Dentro del paquete del clasificador se implementan estas dos clases. Estas recogen toda la lógica necesaria para aplicar el sistema de clasificación a la foto capturada. Para

esto se definen una serie de funciones relacionadas con la implementación y el manejo del modelo utilizado.

A continuación y antes de exponer estos módulos, se definen las dos partes principales por las que se compone este sistema de clasificación:

- *Mobilenet_quant_v1_192.tflite*

  Modelo previamente entenado del tipo *Mobilenet* utilizado para la clasificación. Existen distintos modelos pre-entrenados destinados para ser ejecutados de manera eficiente en dispositivos móviles. Después de realizar distintas pruebas con varios de ellos[24] y de analizar las limitaciones del hardware compatible utilizado para el desarrollo de la aplicación, se ha determinado este modelo como el más adecuado para el sistema de clasificación.

- *Labels.txt*

  Archivo que contiene las etiquetas enteras correspondientes a cada objeto de clasificación.

- Recognition

  Esta clase se utiliza para la creación de objetos que corresponderán con cada resultado. Dentro de esta clase se definen los distintos atributos y métodos por los que estarán formados cada uno de ellos. Estos atributos son:

  - *id*. Identificador único para cada resultado
  - *title*. String que almacena el nombre del resultado
  - *confidence*. Porcentaje de clasificación del resultado según el modelo implementado.

  A continuación se implementa el constructor y los métodos para obtener cada uno de los atributos de cada resultado. Estos métodos sirven como herramientas para la clase *TensorFlowHelper.*

- TensorFlowHelper

  A lo largo de esta clase se definen las funciones necesarias para la clasificación de imágenes a través de TensorFlow. En primer lugar se define un valor entero *RESULTS_TO_SHOW,* con el cual podemos modificar el número de resultados

24 https://github.com/tensorflow/models/blob/master/research/slim/nets/mobilenet_v1.md

que queremos obtener de cada fotografía capturada. Dentro de esta clase se destacan las siguientes funciones.

La función *loadModelFile* es la encargada de abrir el modelo de clasificación utilizado, mientras que la función *readLabels* abre y lee el archivo *txt* de las etiquetas. Ambas devuelven los datos necesario para la clasificación.

Por otro lado se define la función *getBestResults*. Esta es la función principal encargada de obtener los mejores resultados de clasificación. Es en esta dónde se crea una lista de instancias de la clase *Recognition* donde se almacenan los tres resultados con más porcentaje de acierto. Para esto realiza una comparación de los porcentajes calculados para cada etiqueta definida en el *txt*. En esta última parte de la lógica se ha diseñado un filtro para solo ofrecer como resultado aquellas etiquetas con un mínimo de porcentaje.

- Añadir inteligencia artificial al flujo principal

Del mismo modo que antes, para poder hacer uso de estas clases en el flujo principal de la aplicación se implementan unas funciones en la activity *ImageClassidierActivity*. En primer lugar se importa el paquete *Interpreter* de *TensorFlow* Lite. A continuación se definen tanto dos *strings* representando los componentes principales del sistema de clasificación como dos instancias de las clases anteriores.

```
import org.tensorflow.lite.Interpreter;
  […]
private static final String LABELS_FILE = "labels.txt";
private static final String MODEL_FILE ="mobilenet_quant_v1_224.tflite";
  […]
private Interpreter mTensorFlowLite;
private List<String> mLabels;
```

Código 21. Instancias de las clases destinadas al manejo del sistema de clasificación

Finalmente se definen las funciones necesarias para el uso de estas variables. Estas funciones son *initClassifier, destroyClassifier y doRecognize,* definidas en el anexo V.

4.3.4 Formatos de salida

Esta segunda aplicación no hace uso de ninguna interfaz gráfica que proporcione una comunicación con el sistema operativo ni un entorno visual al usuario. Como se ha

mencionado anteriormente en este capítulo, el control de la aplicación se llevará a cabo a través de los periféricos descritos de la Rainbow HAT. En cuanto al entorno visual utilizado para mostrar los distintos resultados al usuario se han diseñado dos opciones de funcionamiento.

- Pantalla HDMI

  Esta primera opción es la descrita en la sección 3.3.3.1. La cual consiste en la utilización de una pantalla dónde se representan el layout denominado *activity_camera.xml* representado en la figura X. Esta opción solo se llevará a cabo cuando se disponga de una pantalla HDMI.

- Display

  Esta segunda opción es otra agregación implementada debido a la necesidad de un entorno visual en las ocasiones en las que no se dispone de una pantalla con conexión HDMI. Esta consiste en la representación de los resultados a través de un periférico de la Rainbow HAT, el display.

  Esta implementación comienza con la importación del paquete correspondiente al *driver* del display y con la la definición de su correspondiente variables y pin GPIO.

```
private static final String I2C_BUS = "I2C1";
private AlphanumericDisplay mSegmentDisplay;
```

Finalmente se implementa la función *display*, la cual crea el objeto display y modifica su estado dependiendo del valor obtenido, llamada al obtener un resultado. También se añade su correspondiente función *onDestroy* para su liberación de recursos.

4.3.5 Compilación y resultados

Similar a la primera aplicación, para obtener el sistema de clasificación final el último paso es la compilación del proyecto en la Raspberry asegurando, previamente, tener conexión con ella y verificando que se puede acceder a dicho hardware a través de los comandos adb.

En cuanto a lo relacionado con el proceso de evaluación de este segundo proyecto no se han podido tener resultados visibles hasta prácticamente finalizada la aplicación. Aún así, se ha utilizado la herramienta de *Debug* de Android y comandos *Log* para depurar la

aplicación a lo largo de todo el desarrollo. Una vez finalizados estos procesos de compilación y evaluación ya se obtiene el sistema de clasificación final. A continuación se exponen algunos de los resultado obtenidos en con ambos diseños del entorno visual.

Resultado 1:

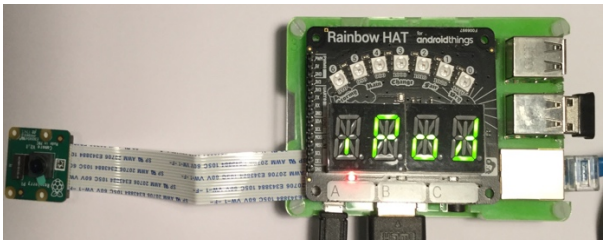
Entorno visual pantalla HDMI

Entorno visual display

Figura 32. Resultado 1 sistema de clasificación

Resultado 2:

Entorno visual pantalla HDMI



Entorno visual display

Figura 33. Resultado 2 sistema de clasificación

Con la realización de este segundo proyecto se ha obtenido el conocimiento sobre cómo integrar Android Things con TensorFlow para diseñar un sistema de clasificación de imágenes. A partir de este desarrollo se pueden realizar numerosas aplicaciones, algunas de estas se encuentran definidas en el capítulo 5.

Capítulo 5

Gestión del proyecto

5.1 Introducción

Debido a la importancia vital que tiene la fase de gestión de cualquier proyecto, en esta sección se detalla una estimación tanto de los recursos económicos y las herramientas necesarias, como de las fases de desarrollo llevadas a cabo.

El primer paso para la gestión de este proyecto es la definición de los distintos objetivos y resultados que se desean obtener con el diseño y el desarrollo del proyecto. Estos objetivos variaran parcialmente según avanza el trabajo realizado. Gracias a la buena realización de este primer paso, estos objetivos se desglosan para generar las distintas fases de desarrollo, las que permiten la realización del cálculo de los costes y recursos utilizados.

5.2 Fases de desarrollo

Para la realización de este trabajo y para poder llegar a los resultados obtenidos se han llevado a cabo numerosas actividades. A continuación, estas actividades se clasifican en distintas fases claramente diferenciadas:

- **FASE 1:** Estudio y documentación

Esta primera fase de desarrollo recogerá todas las actividades relacionadas con la búsqueda de información y con el estudio de todos los conceptos y herramientas empleadas en el proyecto.

Este estudio comienza con un análisis del estado del arte del Internet de las cosas consultando distintos artículos y documentos tratando temas desde su inicio hasta su posición actual. Esta fase también incluye una introducción al sistema operativo de Android Things, estudiando desde los conceptos más básicos hasta su arquitectura. A continuación se realiza una investigación de los distintos servicios que nos proporcionan las dos herramientas utilizadas en las aplicaciones, *Firebase* y *Tensorflow*.

Por último añadir que, aunque la mayor parte de esta fase tenga lugar al inicio del proyecto, debido a la necesidad del mismo de la actualización constante de información acerca de las distintas tecnologías empleadas, también está presente a lo largo del resto de las fases.

- **FASE 2:** Despliegue del entorno de trabajo

Antes de comenzar con la implementación de las distintas aplicaciones se procede a la obtención y el despliegue del entorno de trabajo. Se comienza con la instalación del entorno de desarrollo integrado oficial para la plataforma Android.

Para el desarrollo del proyecto la Universidad Carlos III de Madrid proporcionará un Kit de desarrollador denominado *Raspberry Pi 3 Starter Kit* dónde se compilan las distintas implementaciones. Por otro lado, para el desarrollo de la aplicación del clasificador de imágenes, se adquiere el módulo de cámara oficial de *Raspberry Pi, Camera V2*.

La Universidad Carlos III habilita un espacio situado en el edificio cuatro del campus de la Universidad Carlos III de Leganés, cuya ubicado se encuentra en el 4.1.F02. Este espacio cuenta con varias herramientas con el objetivo de llevar a cabo las reuniones entre los tutores y el alumno para la evaluación y el análisis de resultados. Estas herramientas son una pantalla HDMI y conexión a Internet.

Para finalizar con la preparación del entorno de trabajo se prepara el hardware obtenido para comenzar con el diseño de los proyectos. Como se ha definido anteriormente, esta preparación consiste en la instalación de la imagen del sistema *Android Things* en dicho hardware, debido a las actualizaciones de dicho sistema este paso ha sido repetido varias veces con diferentes herramientas.

- **FASE 3:** Documentación para programar aplicaciones para *Android Things*

Una vez adquiridos la mayor parte de conocimientos necesarios y realizado el despliegue del entorno de trabajo se obtienen y se prueban distintas aplicaciones básicas de Android Things proporcionadas por Android. Estas aplicaciones irán desde el control

local de los periféricos más sencillos hasta la utilización de algunas de las dependencias de *Tensorflow*.

Esta fase, gracias a la variedad de explicaciones que ofrece Google y a realizar constantes experimentos a partir de estas pequeñas aplicaciones, proporciona el conocimiento y el código necesario para poder iniciar los desarrollos de las aplicaciones finales.

- **FASE 4:** Diseño y desarrollo de las aplicaciones

A raíz de algunos de los ejemplos proporcionados por la página oficial de *Android Developers* se procede a la implementación de ambas aplicaciones buscando aquellos objetivos definidos previamente. En esta fase es importante añadir que a lo largo del desarrollo de ambas aplicaciones, estos objetivos fijados que se deseaban obtener con estos desarrollos fueron actualizándose a medida que lo hacía la tecnología y surgían mas posibilidades.

Finalmente añadir que a lo largo de las distintas implementaciones se realizaron distintas consultas a varias comunidades de Google destinadas a este tipo de tecnologías.

- **FASE 5:** Redacción de la memoria

La redacción de la memoria, al igual que la primera fase, se lleva a cabo a lo largo del resto de fases ya que desde el inicio del proyecto se ha ido recopilando y organizando información para su posterior redacción.

Dentro de la memoria se pueden diferenciar distintos capítulos ya definidos en el apartado 1.4.

- **FASE 6:** Preparación de la defensa del proyecto

Para la realización de esta última fase se ha realizado un resumen del proyecto para obtener los puntos más relevantes del mismo.

5.3 Planificación

En la tabla que se muestra a continuación se detalla un resumen del tiempo aproximado en horas de cada una de las fases del desarrollo. También se muestra un desglose de las distintas actividades que conforman cada una de las fases.

FASE	NOMBRE	HORAS
1	**Investigación y documentación**	**200**
1.1	Lectura e investigación del estado del arte y de las distintas tecnologías utilizadas	110
1.2	Estudio e investigación de la plataforma de Android Things y su hardware compatible	90
2	**Despliegue del entorno de trabajo**	**21**
2.1	Obtener todo el Hardware necesario	4,5
2.1	Montar kit de desarrollador	0,5
2.2	Despliegue del entorno de trabajo en el espacio habilitado por la Universidad Carlos III	2
2.3	Documentación sobre configuración del Hardware	3
2.4	Crear producto en Android Things. Descargar imagen y descargar SDK Platform-Tools para Mac	2
2.5	Flashear la imagen	5
2.6	Conectar Hardware y descargar Android Things Setup Utility	4
3	**Documentación para programar aplicaciones para Android Things**	**9**
3.1	Investigación sobre implementaciones de Android Things	8
3.2	Descargar ejemplos para comenzar los desarrollos	1
4	**Diseño y desarrollo de aplicaciones**	**250**
4.1	Desarrollo de Android Things con Firebase	100
4.1.1	Creación cuenta Firebase	2
4.1.2	Desarrollo de la aplicación de la Raspberry Pi	40
4.1.3	Desarrollo de la aplicación móvil	23
4.1.4	Pruebas	35
4.2	Desarrollo de Android Things con TensorFlow	150
4.2.1	Obtener modelo *ImageNet* pre-entrenado	10
4.2.2	Obtener e integrar el módulo de la cámara	20
4.2.3	Modificaciones para la agregación de inteligencia artificial	60
4.2.4	Modificaciones para agregar la cámara al desarrollo	20
4.2.5	Pruebas	40
5	**Redacción de la memoria**	**160**
5.1	Definir la estructura y los puntos de desarrollo	10
5.2	Redacción de la memoria	150
6	**Preparación de la defensa del proyecto**	**40**

Tabla 1. Duración de las fases del proyecto en horas

Se recuerda que los procesos de investigación y de pruebas se han llevado a cabo a lo largo de todas las fases del proyecto. A la duración total del proyecto se añade 3 horas aproximadas correspondientes a las distintas reuniones de inicio y de seguimiento del proyecto con los tutores. Con estos datos y junto con el desglose anterior de las distintas fases del proyecto se determina que la realización del mismo tiene una duración total de aproximadamente 683 horas.

Para facilitar la compresión de este proceso se adjuntará un diagrama de Gantt donde se representa de una manera mas gráfica tanto las distintas actividades de cada fase con su duración, como la duración aproximada total del proyecto. Este diagrama se encuentra en el anexo III.

Junto con el desarrollo de este proyecto se compaginan otras actividades lo cual se requiere una aún mejor organización para una correcta distribución de los recursos. Por otro lado, la fecha inicial establecida para la presentación del proyecto se aplaza de julio a octubre debido a esta reorganización.

5.4 Recursos empleados

Previo a la estimación de los costes totales que suponen la realización de este trabajo se realiza una enumeración de los recursos empleados en la realización de este proyecto. Estos se clasifican en tres tipos:

- Recursos software
    - Sistema operativo del MacBook Pro: Apple OS X 10.11.6
    - Sistema operativo del Xperia Z3 (D6600). Versión Android 6.0.1
    - Android Studio
    - SD Card Formatter
    - SDK Platform-Tools para Mac
    - Android Things Setup Utility
    - Imagen de Android Things Console
    - Modelo clasificador de TensorFlow: MovileNet_v1
- Recursos hardware
    - Ordenador portátil MacBook Pro 12,1
    - Teléfono móvil Sony Xperia Z3 (D6600).
    - Kit de desarrollador: *Raspberry Pi 3 Starter Kit*
        - Raspberry Pi 3
        - Rainbow HAT Project Board
        - Special Edition Pibow Case
        - 2.5ª Worldwide Power Supply
        - 8GB microSD Card
    - Camera Board V2
    - Pantalla HDMI

- Otros recursos

  - Cable USB de Android
  - Cable HDMI
  - Cable de red Ethernet RJ45
  - Conexión Wifi

4.5 Presupuesto

En esta sección, una vez detallado la duración de las distintas actividades y listado los recursos utilizados, se expone una estimación del presupuesto global del proyecto. Este presupuesto se divide en dos partes claramente diferenciables: coste personal y coste material.

- Coste personal

En cuanto a los costes de personal primero definiremos los perfiles que han participado en el proyecto con una estimación del salario por hora de cada uno de ellos en función de su perfil y de un análisis del sueldo que establecen distintas empresas españolas:

Ambos tutores del proyecto: Dos ingenieros superiores cuyos salarios medios se estiman para un ingeniero senior de 23 euros la hora.

Alumno: Desarrollador del proyecto, finalizando el Grado en Ingeniería de Sistemas Audiovisuales cuyo salario medio se estima para un ingeniero medio de 13 euros la hora.

Nombre	Categoría	Horas estimadas	Sueldo	Coste
Mª Celeste Campo Vázquez	Ingeniero Senior	30	23€/hora	690€
Carlos García Rubio	Ingeniero Senior	30	23€/hora	690€
David Gonzáles Ramos	Ingeniero Junior	680	13€/hora	8840€
TOTAL				**10220€**

Tabla 2. Estimación coste personal

- Coste material

Para el cálculo del coste material total analizaremos tanto los recursos hardware como el software, definidos en el apartado anterior, estimando sus costes imputables teniendo en cuenta el porcentaje de uso de cada uno. Para realizar esta estimación se hace uso de la ecuación de amortización siguiente:

$$CI = A/B * C * D$$

Ecuación del coste imputable

Donde: CI, coste imputable.
A es el tiempo de uso medido en meses.
B es una estimación de su vida útil.
C corresponde con el coste.
D porcentaje dedicado al proyecto (generalmente 100%)

Recurso	Coste	Vida útil	% de uso	Tiempo de uso	Coste imputable
Laptop MacBook Pro 12,1	1399 €	60 meses	100%	10 meses	233,17 €
Smartphone Sony Xperia Z3 (D6600)	140 €	60 meses	40%	10 meses	9,33 €
Raspberry Pi 3 Starter Kit	80 €	60 meses	100%	10 meses	13,33 €
Camera Board V2	25 €	60 meses	100%	4 meses	1,67 €
Pantalla HDMI	90 €	60 meses	60%	10 meses	9 €
Cable HDMI	8 €	60 meses	60%	10 meses	0,8 €
Cable de red Ethernet RJ45	2 €	60 meses	60%	10 meses	0,2 €
Android Studio	0 €	-	100%	10 meses	0 €
Paquete Microsoft Office	46 €	40 meses	50%	6 meses	3,42 €
Licencia Android	0 €	-	100%	10 meses	0 €
Software destinado a la Raspberry	0 €	-	100%	10 meses	0 €
TOTAL:					**270,92 €**

Tabla 3. Estimación del coste material

- Coste total

Para el cálculo del coste total del proyecto se tiene en cuenta los costes indirecto. Estos costes son los relacionados con la conexión wifi, la tarifa de luz, mejoras de última hora, etc y se estiman como un 20% del los costes del proyecto. En la siguiente tabla se representa el desglose de los distintos costes.

Tipo de coste	Coste total
Coste personal	10220 €
Coste material imputable	270,92 €
Costes indirectos	2098,18 €
I.V.A	2643,71 €
TOTAL	**15232,81** €

Tabla 4. Estimación del coste total

5.6 Entorno socio-económico

Como se ha expuesto en capítulos anteriores, en el mundo globalizado actual estas tecnologías (inteligencia artificial, internet de las cosas, domótica) permiten a los dispositivos electrónicos la capacidad de comunicarse entre ellos sin la intervención de un ser humano. Desde 2009, el número de dispositivos conectados ha superado el número de personas en la Tierra y en la actualidad, el número de dispositivos conectados ha alcanzado los 9 mil millones y se espera que exista un crecimiento muy alto para 2020.

- Impacto socio-económico

El empleo en España se puede caracterizar, desde mediados los años 80, por la alta tasa de desempleo y una elevada proporción de empleo de baja calidad.

Esta tecnología junto con otras, permiten la sustitución de tareas cada vez mas complejas y cognitivas. Algunos estudios señalan que la velocidad de esta sustitución, que conlleva la eliminación de puesto de trabajo, al ser superior a la velocidad de creación de nuevos empleos acabaría provocando desempleo. Por otro lado se estima un incremento en puestos de trabajo relacionados con los servicios de gestión, planificación y relación personal, estos podrían ser la sanidad, la enseñanza, la restauración, etc. Aún así, esta evolución tecnológica del IoT se integrará en estas áreas y provocará modificaciones.

Actualmente, las empresas españolas están realizando una inversión progresiva en esta tecnología, debido a su poder para mejorar la productividad y la eficiencia en sectores como la sanidad, el *retail*, la industria, etc.

- La sanidad es el área dónde los dispositivos conectados tienen mayor futuro. Ahora, multitud de dispositivos conectados entre sí, pueden proporcionar gran cantidad de datos e información médica en tiempo real.

- El sector público ha sido uno de los pioneros en utilizar soluciones de Internet de las cosas, como por ejemplo dispositivos de reconocimiento, analizar el flujo de visitantes, etc.

- En el sector industrial, esta tecnología se aplica en el mantenimiento y en los distintos sistemas de control y automatización. En las fábricas y los entornos industriales evolucionan a la pérdida de los cables.

El hecho es que el uso de dispositivos IoT supone toda una ventaja en lo referente al sector industrial y, más concretamente, cuando son aplicados en el mantenimiento y su uso en los sistemas de control y automatización

- Impacto medio-ambiental

El internet de las cosas puede tener un impacto positivo en el medio ambiente debido a su poder de transmisión de grandes cantidades de datos entre maquinas. Esta tecnología permite transmitir diferentes indicadores ambientales, tales como la radiación, la calidad del agua, los productos químicos peligrosos, ofreciendo a las personas que trabajan en ambientes peligrosos o incluso a pacientes beneficiarse de dicha tecnología al poder conectarse a diferentes dispositivos para enviar o consultar enormes cantidades de datos en la red. Gracias a esto, se puede llegar a tener una compresión mas amplia del entorno actual en el que vivimos permitiendo encontrar diversas soluciones para numerosos problemas ambientales.

Otros aspectos donde el IoT puede beneficiar al medio ambiente es en la reducción de las emisiones de CO2 o en el ahorro de recursos en la agricultura ajustando, mediante distintos sensores, los procesos de producción basados en el clima o la luz solar. Se estima que para 2020 el internet de las cosas podría ahorrar 1,6 Gt de CO2.

El impacto medioambiental causado por la integración del Internet de las cosas no solo tiene consecuencias positivas, también se debe tener en cuenta un gran reto medio ambiental que representan los desechos electrónicos. Debido a la velocidad de desarrollo de esta tecnología en los próximos años existirán millones de dispositivos obsoletos que pueden impactar en el medio ambiente si no se toman medidas.

- Impacto ético [27]

El desarrollo de esta tecnología está afectando cada vez más a todos los aspectos de la vida humana, tanto al comportamiento y a las relaciones sociales como a los valores morales y éticos, generando nuevos desafíos y soluciones relacionados con estas nuevas tecnologías. Este desarrollo acelerado ha generado nuevas posibilidades de desarrollo, sin embargo, debido a estos aspecto de la vida afectados por estas tecnologías, han surgido cuestiones éticas.

El uso de estos nuevos medios tecnológicos puede desembocar en la realización de algunas acciones consideradas desde un punto de vista menos positivo tanto para las personas como para las organizaciones, algunas de estas son:

  - La pérdida de seguridad y privacidad
  - La violación a la propiedad intelectual
  - El monitoreo sin consentimiento de las actividades que desarrolla una persona

En 1999 fue creada la *International Center for Information Ethics* (ICIE) para el debate de distintos problemas éticos relacionados con el Internet y la comunicación digital. Actualmente, ICIE (2018) es una comunidad académica dedicada al avance del campo de la ética de la información [28].

La sociedad actual se ha convertido en un dependiente total de las tecnologías de información y las herramientas tecnológicas. Gracias a estas, el acceso y divulgación de la información ha atravesado barreras geográficas, pero también generan inconvenientes como los delitos informáticos o problemas de privacidad.

A lo largo de este proyecto se ha trabajado con una plataforma para el Internet de las cosas, Android Things, con la cual se han realizado distintos desarrollos a través de los cuales se ha conocido más a fondo esta tecnología y sus posibilidades con el objetivo, entre muchos otros, de entender el entorno socioeconómico en el que se encuentra. Como conclusión de este proyecto y de este capítulo se determina que el Internet de las cosas está cambiando y cambiará por completo el escenario socioeconómico tal y como se conoce.

Capítulo 6

Conclusiones y futuras mejoras

6.1 Conclusiones

Tras la realización tanto de la investigación necesaria para el desarrollo de este proyecto como la implementaciones de las distintas aplicaciones se ha llegado a varias conclusiones. Algunas de estas se destacan a continuación.

Gracias a la investigación sobre el pasado y el presente de las tecnologías IoT y del nuevo sistema operativo de esta tecnología de Google se ha llegado a la conclusión del claro futuro de los dispositivos que conforman este concepto. Este futuro se caracteriza por un aumento exponencial de estos dispositivos con sensores conectados y por lo tanto de un crecimiento de datos digitales masivo.

Por otro lado, gracias a los desarrollos con esta nueva tecnología, Android Things, junto con las demás tecnologías se ha demostrado las numerosas funciones que se pueden realizar dentro de este campo y la infinidad de aplicaciones posibles aún por desarrollar.

En cuanto a los objetivos iniciales de manera general se puede concluir que se han cumplido. Al realizar y obtener los sistemas resultantes que conforman las aplicaciones finales se llega a cumplir el objetivo principal de servir, dichas aplicaciones, como un inicio para futuros desarrollos destinados a la mejora de nuestras vidas. Por otro lado, a lo largo de la realización del proyecto, también se han ido alcanzando los objetivos parciales definidos inicialmente. Sobre el alcance de estos objetivos parciales cabe destacar que gracias al desarrollo del proyecto se ha adquirido un buen conocimiento sobre el sistema operativo Android Things.

Aunque estos objetivos inicialmente planteados han llegado a finalizarse, a lo largo de este trabajo fin de grado han surgido numerosas ideas y desarrollos paralelos con los que continuar este proyecto centrándose tanto en la tecnología principal en la que se basa, Android Things, como en las demás tecnologías usadas. Estas líneas futuras se detallan en la sección 6.2.

A lo largo de todo el proyecto se ha hecho uso tanto de conocimientos ya adquiridos ya sea por parte de la universidad o por otros medios como de conocimientos que se han ido adquiriendo a lo largo de la investigación y el desarrollo del mismo.

En cuando al conocimiento relacionado con los desarrollos de los módulos Android Things implementados en la Raspberry ha sido adquirido totalmente durante el proyecto a través de su página web oficial, foros destinados a esta tecnología y con la realización de pruebas y la evaluación de resultados. Sobre las implementaciones relacionadas únicamente con Android no ha surgido ningún problema al tratarse de desarrollos sencillos basados en el conocimiento adquirido en la carrera.

Finalmente destacar que tanto en la parte de configuración de la Raspberry Pi como en la de el manejo de imágenes en la aplicación de Android Things y TensorFlow han sido las partes del proyecto donde más dificultades y retrasos han surgido.

- Configuración de la Raspberry Pi. En concreto se dedicó bastante tiempo a la instalación del sistema operativo de Android Things en la Raspberry debido al poco tiempo que llevaba esta tecnología y a la poca información publicada.
- Tratamiento de imágenes en la aplicación de Android Things y TensorFlow. Este problema estuvo relacionada con encontrar el formato adecuado para el tratamiento de las imágenes a través del sistema de clasificación.

6.2 Futuras mejoras

Después de analizar las conclusiones de este proyecto se pretende, en esta sección, detallar algunas líneas futuras en las que el proyecto y las distintas tecnologías podrían mejorar los resultados obtenidos.

Como se ha mencionado, a lo largo de la realización del proyecto y al trabajar con tecnologías innovadoras y en constante actualización, han surgido tanto numerosas ideas y mejoras como diferentes desarrollos en relación con las distintas aplicaciones diseñadas. Por otro lado, estas ideas se han incrementado con la comunicación establecida entre numerosos usuarios de estas tecnologías a través de las comunidades de Google. A continuación, estas ideas futuras se han clasificado en dos aspectos: futuras mejoras para

las distintas aplicaciones desarrolladas y futuras mejoras a través de las tecnologías utilizadas.

- Futuras mejoras de las aplicaciones:

    - Aplicación Android Things junto con Firebase. Respeto a esta aplicación existen infinidad de mejoras posibles para implementarse en el futuro. Comenzando con la aplicación móvil se puede diseñar una interfaz mas compleja con un control más extenso sobre un mayor número de periféricos de la Raspberry. En cuanto al siguiente paso del sistema de control completo sería su integración con cualquier entorno de producción como por ejemplo una caldera de un hogar o un sistema domótico de luces.

    - Aplicación Android Things junto con TensorFlow. Respecto a este proyecto existen numerosos campos para su aplicación en diferentes entornos de producción. Según estos se pueden desarrollar una serie de mejoras centradas en sus objetivos. Uno de estos ejemplos trataría sobre su integración en el campo de la medicina, ayuda para personas invidentes. Algunas mejoras para este caso sería la integración de un altavoz como periférico o una integración parcial con el proyecto de Android Things con Firebase para capturar imágenes de forma remota.

- Futuras mejoras a través de la tecnología usada:

    A lo largo de este punto se detallaran las posibles mejoras no tanto en relación con la aplicación en sí, sino con las distintas tecnologías utilizadas en los desarrollos.

    En cuanto a la tecnología en la que se base este proyecto, Android Things, se ha visto a lo largo del proyecto las numerosas posibilidades que ofrece y la amplia gama de aplicaciones en las que se puede incluir.

    Con respecto a Firebase, a parte del servicio de RealTime Database utilizado en la aplicación, esta tecnología ofrece muchos mas servicios con posibilidades de desarrollar numerosos casos de usos. Algunos de estos posibles casos de uso son los siguientes.

    - Diseñar un flujo de incorporación satisfactorio. Con el uso de los servicios de Firebase de *Authentication* y *Remote Config* se puede diseñar una aplicación de control remoto cuyo acceso sea a través de una cuenta de Facebook, Twitter, Google, etc.

- Personalizar una pantalla de bienvenida para cada usuario a través de los servicios de *Remote Config* y *Analytics.*
- El diseño de un sistema para seguir el recorrido de los usuarios por distintos dispositivos para obtener mas datas y comprender mejor al usuario. Para esta mejora se utilizan los servicios de *Big Query* y *Analytics*.

Por último, otra de las más cercanas y lógicas mejoras que se pueden realizar en relación con la tecnología de Tensorflow, es el reentrenamiento propio de un modelo de clasificación a partir de un *datasheet* que nos interese para que, junto con el sistema operativo del internet de las cosas Andorid Things, se diseñe un sistema de clasificación y de detección mucho mas personalizado. Este entrenamiento de un modelo propio se realizaría a partir de los parámetros de algún modelo moderno de reconocimiento reutilizando las capacidades de extracción de características del mismo.

Bibliografía

[1] "Historia de Internet –nacimiento y evolución". En línea. Septiembre 2013. Disponible en: http://redestelematicas.com/2013/09/

[2] "Un poco de historia sobre IoT". En línea. Abril, 2012. Disponible en: http://www.sorayapaniagua.com/2012/04/15/un-poco-de-historia-sobre-internet-de-las-cosas/

[3] "Internet of Things – essential IoT business guide". Disponible en: https://www.i-scoop.eu/internet-of-things-guide/

[4] Sitio oficial Firebase: https://firebase.google.com/

[5] Sitio oficial TensorFlow: https://www.tensorflow.org/

[6] Sitio oficial de Android: https://www.android.com/

[7] Burnette, E. (2015). "Hello, Android: introducing Google's mobile development platform". Pragmatic Bookshelf.

[8] Sitio oficial Android, "Kotlin": https://developer.android.com/kotlin/

[9] Sitio oficial de Android Developers: https://developer.android.com/

[11] Gladys Dapozo, Patricia Pesado, Emanuel Irrazabal "XX Workshop de Investigadores en Ciencias de la Computación - WICC 2018". En línea. Corrientes : Universidad Nacional del Nordeste. Facultad de Ciencias Exactas, 2018. Disponible en: http://wicc2018.unne.edu.ar/wicc2018librodeactas.pdf

[12] Sitio oficial Google Cloud Plataform: https://cloud.google.com/?hl=es

[13] Sitio Oficial de Android Things, "Descripción general de Andorid Things Console" https://developer.android.com/things/console/

[14] A. Cama Pinto, E. De la Hoz Franco, y D. Cama Pinto, Las redes de sensores inalámbricos y el internet de las cosas, INGE CUC, vol. 8, n.º 1, pp. 163-172, oct. 2012.

[15] Rodrigo Martinez Jacobson, "Comparativa y estudio de plataformas IoT" Trabajo Fin de Grado, Departamento de Informática, Universitat politécnica de Catalunya, 2017.

[16] Manuel Zaforas, "TensorFlow, o cómo será el futuro de la Inteligencia Artificial según Google". En línea. 31 de mayo de 2017. Disponible en: https://www.paradigmadigital.com/dev/tensorflow-sera-futuro-la-inteligencia-artificial-segun-google/

[17] "Resumen del TensorFlow Dev Summit". En línea. 30 de Marzo 2018. Disponible en: https://medium.com/ai-learners/tensorflow-dev-summit-resumen-3e77862f5416

[18] Sitio oficial de TensorFlow, "TensorFlow Hub". Disponible en:
https://www.tensorflow.org/hub/

[19] Ley Orgánica15/1999, de 13 de diciembre, de Protección de Datos de Carácter Personal. Disponible en:
https://www.boe.es/boe/dias/1999/12/14/pdfs/A43088-43099.pdf

[20] Reglamento (UE) 2016/679 del Parlamento Europeo y el Consejo de 27 de abril de 2016 relativo a la protección de las personas físicas en lo que respecta al tratamiento de datos personales y a la libre circulación de estos datos y por el que se deroga la Directiva 95/46/CE. Disponible en
https://eur-lex.europa.eu/legal content/ES/TXT/HTML/?uri=CELEX:32016R0679&from=ES

[21] Directiva 2014/53/UE del Parlamento Europeo y del Consejo de 16 de abril de 2014 relativa a la armonización de las legislaciones de los Estados miembros sobre la comercialización de equipos radioeléctricos. Disponible en:
https://www.boe.es/doue/2014/153/L00062-00106.pdf

[22] Directiva 2013/40/UE del Parlamento Europeo y del Consejo del 2013 relativa a los ataques contra los sistemas de información. Disponible en:
https://www.boe.es/doue/2013/218/L00008-00014.pdf

[23] Comunicación de la comisión al parlamento Europeo, Al Consejo, al Comité Económico y Social Europeo y al Comité de La regiones. Disponible en:
http://www.ciencia.gob.es/stfls/MICINN/Internacional/FICHEROS/Actuaciones_Europeas/Comunicacion_Union_por_la_Innovacion.pdf]

[24] Iniciativa de comunidades conectadas. En línea. 18 de mayo de 2015. Disponible en:
https://ec.europa.eu/digital-single-market/en/news/connected-communities-initiative]

[25] Página Oficial Android Things "Raspberry Pi 3". Disponible en:
https://developer.android.com/things/hardware/raspberrypi

[26] Sitio Oficial Android Things, "Peripheral I/O". Disponible en:
https://developer.android.com/things/sdk/pio/

[27] Eurípides Triana Tacuma, "Ética Informática", En línea. Diciembre 2017, Disponible en:
http://www.spentamexico.org/v12-n3/A17.12(3)272-279.pdf

[28] Sitio oficial del ICIE, Disponible en:
https://www.i-c-i-e.org/home

[29] Arturo González García, "IMPACTO MEDIOAMBIENTAL DE LA INTEGRACIÓN DE LA COMPUTACIÓN EN LA NUBE Y LA INTERNET DE LAS COSAS", 2016.

ANEXO I. Glosario

Manifiesto de la aplicación: Archivo que proporciona información esencial sobre tu aplicación al sistema Android, información que el sistema debe tener para poder ejecutar el código de la app.

SoM: Sistemas de producción en módulos.

WSN: Wireless Sensor Networks

M2M: Machine to machine

ART: Android Runtime

AOT: Ahead of time.

OEM: Original Equipment Manufacturer.

OTA: Actualización por el aire de un dispositivo móvil.

BACK-END: Parte del desarrollo web que se encarga de que toda la lógica de una página web funcione.

PWM: Modulación por anchos de pulsos.

Pantallas LCD: Pantalla delgada que está formada por un determinado número de píxeles que se colocan delante de una fuente de luz.

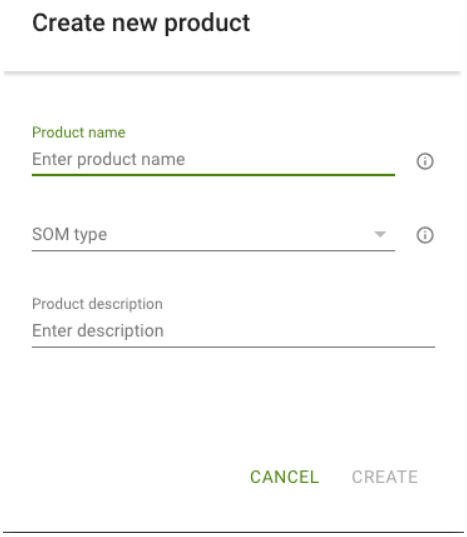
Machine Learning: Es una rama de la Inteligencia Artificial que se dedica al desarrollo de las técnicas necesarias para implementar el aprendizaje automático de las máquinas.

Redes neuronales: Modelo computacional que intenta imitar el comportamiento de las neuronas de cerebros biológicos.

ANEXO II: Manual de administración de productos Android Things

En este anexo se recoge un breve resumen de la utilización de *Android Things Console* para la administración de uno o varios productos.

Para comenzar se deberá abrir la consola de Android Things y registrarse a través de una cuenta Google si es un usuario nuevo. El primer paso será crear un producto Android Things donde se requerirá la siguiente información:



- Nombre del producto
- Plataforma hardware (SOM) en la que se está creando el producto
- Descripción del producto

Imagen X. Crear un nuevo producto

El siguiente paso es la configuración de del producto. Android Things le permite la cambiar la configuración de cada producto, crear diferentes modelos de cada producto y compartirlos con otros usuarios.

La creación de modelos sirve para crear diferentes variantes de compilaciones de las aplicaciones que se ejecutan en un mismo dispositivo. Con la creación de un nuevo producto se genera automáticamente un identificador único de tu modelo, el cuál se puede modificar o crear otro.

Otra opción que permite *Android Things Console* es la de compartir tus productos con otros usuarios de Android Things. Esta acción se realizará con la creación de un grupo de Google.

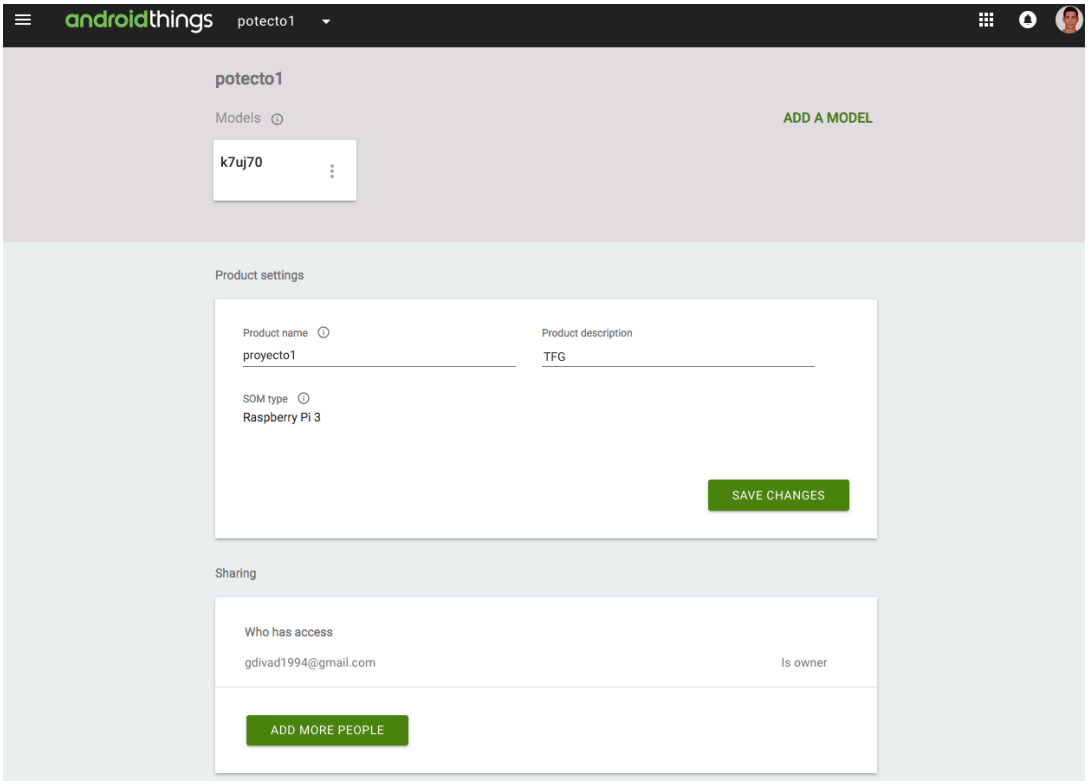


Imagen X. Pantalla de configuración de tu producto

A continuación se se accede a través del ID del modelo a la pantalla de creación de una compilación para dicho modelo.

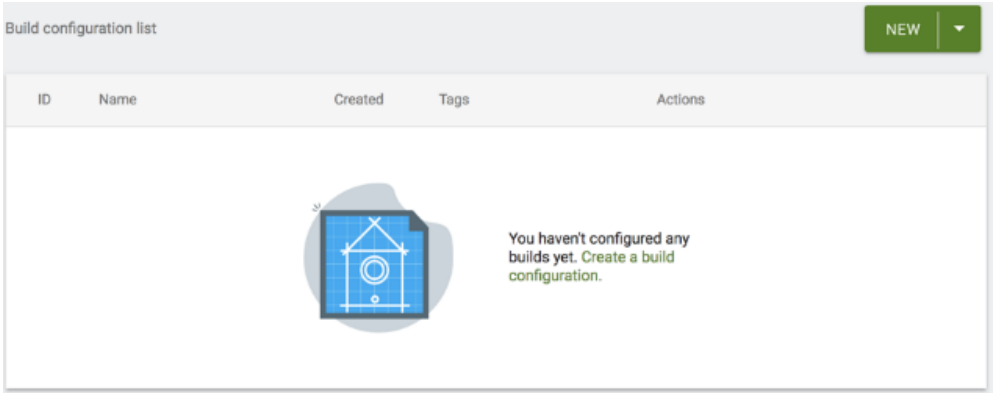


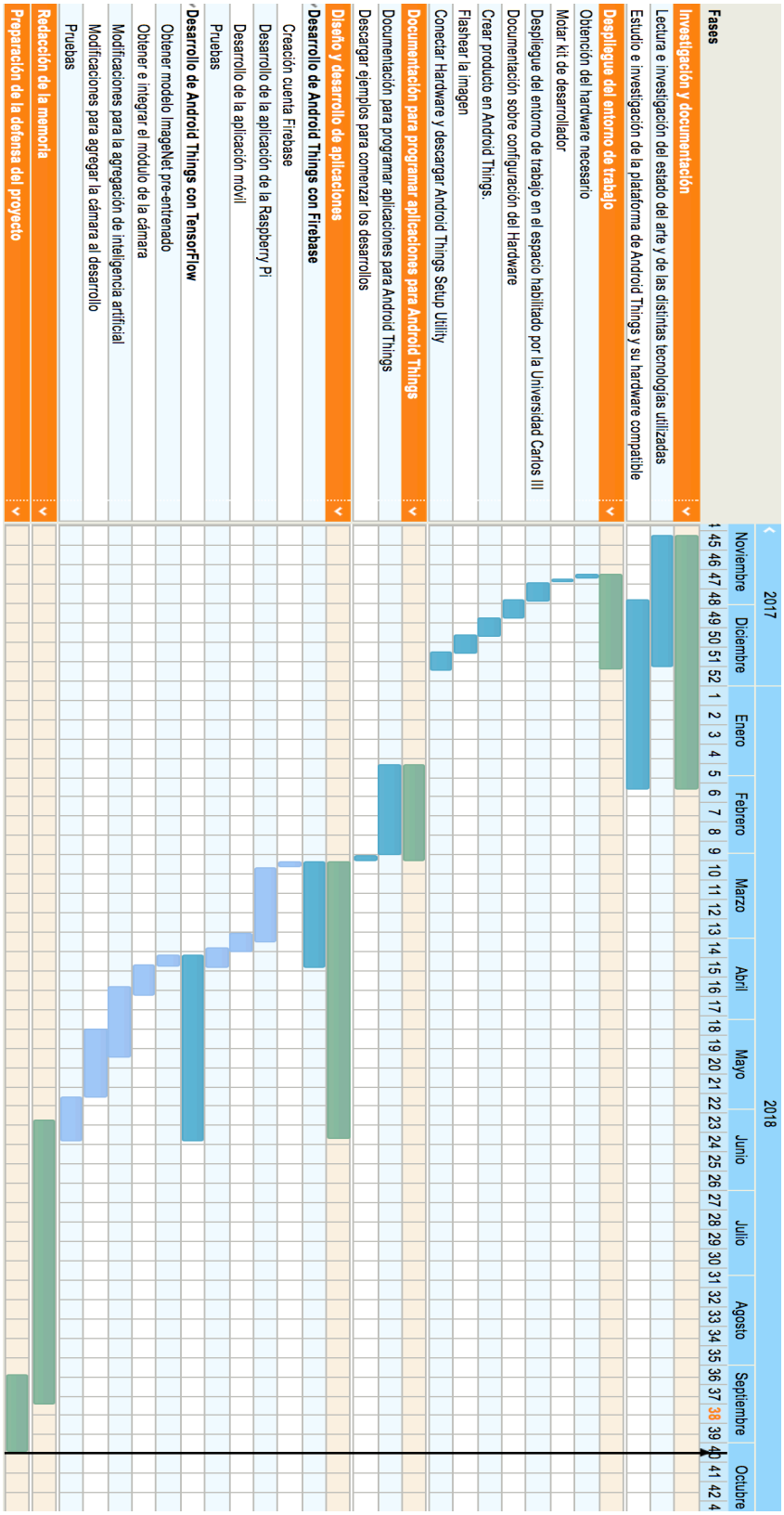
Imagen X. Lista de configuración de compilaciones vacía

Para la creación una construcción para tu producto de Android Things se debe acceder a “nuevo” e introducir la siguiente información: nombre y versión del sistema operativo para su compilación, las aplicaciones que se desea incluir, agregar recursos de compilación adicionales y agregar un nuevo periférico (opcional) para poder adaptarse a todas las aplicaciones.

Al finalizar, aparecerá la compilación creada en la lista. Desde aquí, se podrá etiquetarla o descargarla como imagen de desarrollo o de producción.

Imagen X. Compilación creada

ANEXO III. Diagrama de Gantt



Puntos a destacar sobre el diagrama de Gantt:

- Como se puede observar algunas fases del desarrollo del proyecto coinciden en el tiempo debido que se han llevado a cabo en paralelo.
- Para la representación de este diagrama se ha tenido en cuenta los periodos de mayor peso de cada tarea, ya que algunas tareas han sido desarrolladas durante todo el proyecto.

ANEXO IV. Función inicialización cámara

Este anexo recoge el código correspondiente con la función *initializeCamera.* Dentro de esta función se sigue el siguiente flujo.

En primer lugar se comprueba si se ha llamado anteriormente, es decir, comprueba si la cámara ya está inicializada. Posteriormente se captura la instancia de la cámara y se le intenta asignar sus recursos.

Finalmente se intenta abrir la cámara con los recursos asignados imprimiendo un error si devuelve una excepción.

```
public void initializeCamera(Context context, int previewWidth, int
previewHeight,Handler backgroundHandler, ImageReader.OnImageAvailableListener
imageAvailableListener) {

    if (initialized) {
        throw new IllegalStateException(
           "CameraHandler está inicializando");
    }
    initialized = true;

    CameraManager manager = (CameraManager)
                    context.getSystemService(Context.CAMERA_SERVICE);

    String[] camIds = null;

    try {
        camIds = manager.getCameraIdList();
    } catch (CameraAccessException e) {
        Log.w(TAG, "Cannot get the list of available cameras", e);
    }
    if (camIds == null || camIds.length < 1) {
        Log.d(TAG, "No cameras found");
        return;
    }
    Log.d(TAG, "Using camera id " + camIds[0]);

    mImageReader = ImageReader.newInstance(previewWidth,
                     previewHeight, ImageFormat.JPEG, MAX_IMAGES);

    mImageReader.setOnImageAvailableListener(imageAvailableListener,
                                                backgroundHandler);

    try {
     manager.openCamera(camIds[0], mStateCallback, backgroundHandler);
    }
    catch (CameraAccessException cae) {
        Log.d(TAG, "Camera access exception", cae);

    }
}
```

ANEXO V. Agregaciones

A lo largo de este anexo se detallan las distintas funciones más relevantes de aplicación Android Things con TensorFlow:

- Función *initCamera*. Dentro de esta función se inicializan las dos instancias definidas, *mImagePreprocessor* y *mCameraHandler* Por otro lado se llama *OnImageAvailableListener* donde se invoca *preprocessImage()* el cual pasa el Bitmap procesado a *onPhotoReady(),* lo que lo redirigirá al método de reconocimiento definido anteriormente.

```
private void initCamera() {
    mImagePreprocessor = new ImagePreprocessor(PREVIEW_IMAGE_WIDTH,
PREVIEW_IMAGE_HEIGHT,TF_INPUT_IMAGE_WIDTH, TF_INPUT_IMAGE_HEIGHT);
    mCameraHandler = CameraHandler.getInstance();
    mCameraHandler.initializeCamera(this,
            PREVIEW_IMAGE_WIDTH, PREVIEW_IMAGE_HEIGHT, null,
            new ImageReader.OnImageAvailableListener() {
                @Override
                public void onImageAvailable(ImageReader imageReader) {
                    Bitmap bitmap =
mImagePreprocessor.preprocessImage(imageReader.acquireNextImage());
                    onPhotoReady(bitmap);
                }
            });
}
```

- Función *closeCamera*. Función destinada a la liberación de los recursos reservados para la cámara.

```
private void closeCamera() {
    mCameraHandler.shutDown();
}
```

- Función *loadPhoto*. Encargada de llamar a la función de *CameraHandler* destinada a capturar una imagen.

```
private void loadPhoto() {
    mCameraHandler.takePicture();
}
```

- Función *initClassifier*. Dentro de esta función se inicializan las dos instancias definidas, mTensorFlowLite y mLabels, inicializando con esto el proceso de clasificación. Para ello se leen el archivo del modelo de Tensorflow y la list de etiquetas que mapean los resultados con ayuda de las funciones de *loadModelfile* y *realLabels*.

```
private void initClassifier() {
    try {
        mTensorFlowLite = new Interpreter(TensorFlowHelper.loadModelFile(this,
MODEL_FILE));
        mLabels = TensorFlowHelper.readLabels(this, LABELS_FILE);
    } catch (IOException e) {
        Log.w(TAG, "Unable to initialize TensorFlow Lite.", e);
    }
}
```

- Función *destroyClassifier*. Función destinada a la liberación de los recursos reservados para el clasificador.

```
private void destroyClassifier() {
    mTensorFlowLite.close();
}
```

- Función *doRecognize*. Este es el método que se invoca cuando el usuario solicita una clasificación. Se le pasa la imagen como parámetro y ejecuta el *onPhotoRecognitionReady* , método encargado de dar formato y de mostrar la lista de *Recognition*, es decir, la lista de los resultados.

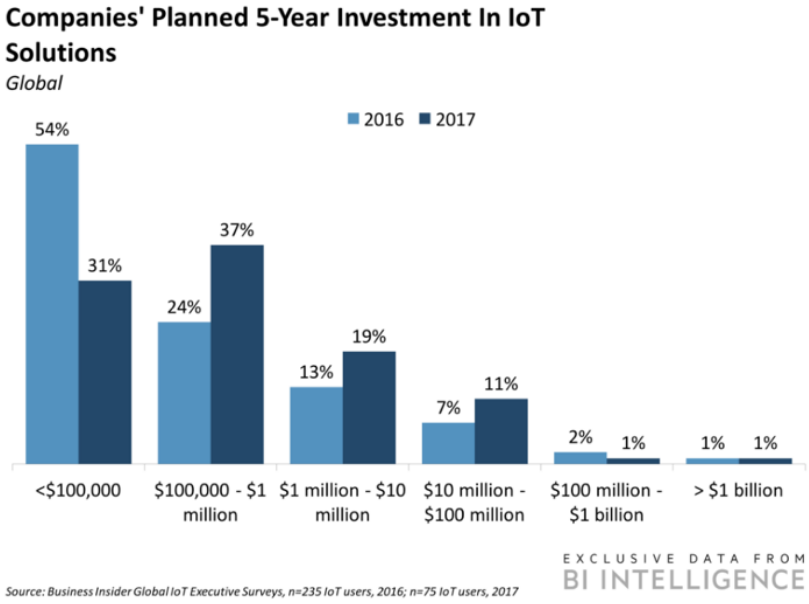
```
private void doRecognize(Bitmap image) {

    byte[][] confidencePerLabel = new byte[1][mLabels.size()];

    int[] intValues = new int[TF_INPUT_IMAGE_WIDTH * TF_INPUT_IMAGE_HEIGHT];
    ByteBuffer imgData = ByteBuffer.allocateDirect(
            DIM_BATCH_SIZE * TF_INPUT_IMAGE_WIDTH * TF_INPUT_IMAGE_HEIGHT *
DIM_PIXEL_SIZE);
    imgData.order(ByteOrder.nativeOrder());
    TensorFlowHelper.convertBitmapToByteBuffer(image, intValues, imgData);
    mTensorFlowLite.run(imgData, confidencePerLabel);
    Collection<Recognition> results =
       TensorFlowHelper.getBestResults(confidencePerLabel, mLabels);

    onPhotoRecognitionReady(results);

}
```

ANEXO VI. Extended Abstract

1.1 Introducción

You could define IoT (Internet of Things), as a term results in MIT (Massachusetts Institute of Technology) this term involve the interconnection of objects just as objects or people through a global network. This way of connecting anything to Internet, turn out the creation of infinite applications, which has meant a new revolution in the world of technology. This concept is endowed with connectivity to objects, developing platforms for them to communicate with each other and with humans to manage the large volume of data that these Things from Internet generate.

Large and Small companies and medium-sized enterprises (SMEs) aim to be favored by the IoT in such a way that during these years it will be one of the axes in which it will be worked by them. Internet of Things is transforming the lifestyle of These companies and global consumers. The technology that Underlying this concept is evolving rapidly.



Investments planned by companies in IoT technologies and solutions

Some data according to the IoT Report made by "Business Insider" are:

There will be more than 55 billion IoT devices by 2025, compared to one billion in 2017 and almost $ 5 trillion in investment added of IoT between 2017 and 2025.

This technology also has a negative side. With its growth, the volume of data and information has increased exponentially and at the same time has multiplied vulnerabilities and innate dangers to any new technology. This is a consecuence of not used standard by manufacturers that have been implementing the Internet , because each one has employed different communication protocols, systems, etc.

Throughout this project we have made a study of Android Things, a new Google' s platform, through the combination with other technologies in different developments. Its potential is also valued with a comparison with other platforms and studying the diversity of possibilities it offers.

1.2 Main objectives

This project has two main objectives that come together in one. On the one hand, as mentioned above, it is the design and development of two projects based on the Google Android Things platform. Another of the general objectives is the introduction to the study and the development of services provided by other usual Google technologies. In particular, we study the Realtime Database service provided by Firebase and a lightweight TensorFlow solution for mobile and integrated devices, TensorFlow Lite. This involves a machine learning tool knows by its efficiency working with neural networks and distributed processes.

Both purposes will have as objective to be useful as a basis for future developments and improvements for the design of applications that allow satisfying the demands of consumers.

1.3 Android Things

Android Things is a platform that come from Android Mobile, it makes easy to developers create and integrate the Android development ecosystem, including APIs and development tools and manufacturers to produce IoT devices. Within this platform we can differentiate four important parts:

- Its operating system as an Android variant
- Architecture
- Wide Hardware
- Developer Console to manage devices and send updates.

There are some important differences between the development of projects for Android Things, since they are optimized for integrated devices and the development of applications in Android Mobile, some of them are the following:

- There are different Android functions and APIs that are currently not compatible with Android Things devices.
- Android Things devices are optimized for the use of a single application that starts automatically at the beginning of the system.
- These embedded devices usually have memory limitations, so the platform requires applications to keep the native libraries within the APK at run time.
- Android Things devices have more flexible access to hardware peripherals and controllers.

1.4 Implementation of Android Things projects

1.4.1 Android Things with Firebase

The main objective of this first project is the introduction in the design of applications for Android Things. Besides from the main objectives of the work, the design and development of this first application has the following particular objectives:

- Acquire the necessary knowledge for the control and management of the different GPIO pins of the Raspberry Pi.
- Learn to integrate the Firebase library in different Android projects, both in mobile devices and in the compatible hardware of Android Things.
- Achieve the necessary knowledge for the use in RealTime Database service provided by Firebase

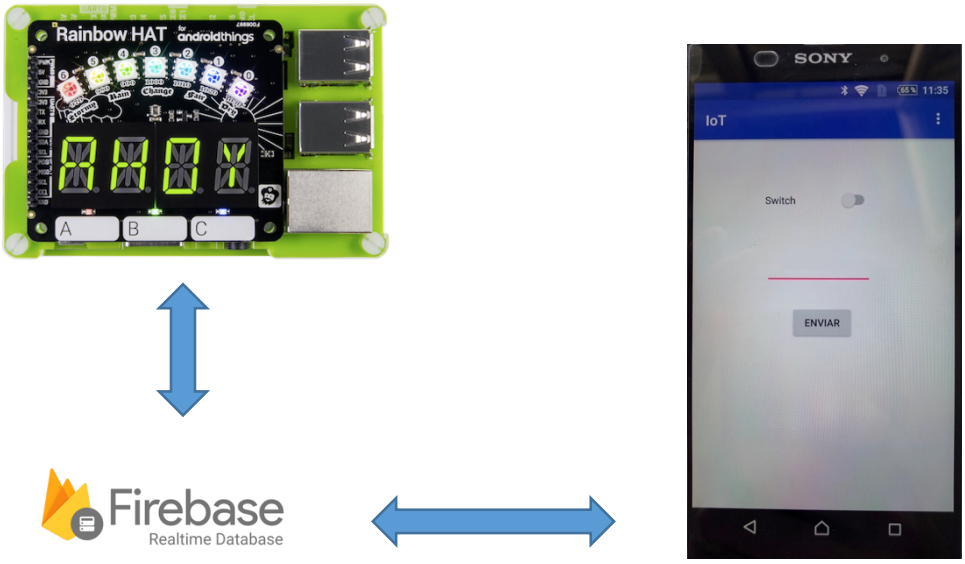
In order to begin its development, an example provided by Android is used through its website. It shows how to use an UserDriver input button, listen for GPIO pin changes and generate events to change the state of a LED. Throughout the development of this application, numerous modifications and additions have been made to this example until achieving the desired final functionality. The most important are highlighted below:

- Complete integration of the project with Firebase, adding the dependencies and the necessary files.
- Development of the necessary logic for the control and management of a LED and the display.
- Development of a mobile application for the control of several peripherals of the Raspberry.

In order to carry out the development of these improvements, the following fundamental requirements must be taken into account:

- Mobile device with Android operating system (V. 6.0.1)
- Android Studio 2.0 or higher
- Project in Firebase
- WiFi connection

Regarding the final design of this first project consists of an application for the Raspberry integrated with Firebase, where the use of the GPIO pins corresponding to a led and the display is defined. Another application for a mobile device, also integrated with Firebase, has an interface that offers control of these pins. The following image shows the scheme of the application.



Architecture of the first application

1.4.2 Android Things with TensorFlow

This second application consists in introduce a Machine Learning engine to an IoT system, that is, the integration of TensorFlow in Android Things. Thanks to this second development, it is possible to evaluate the potential of the combination of both technologies in a final solution. This final solution consists of executing the inference of TensorFlow Lite for Android, creating a device capable of capturing images through a camera and classifying them locally through a previously trained model.

Apart from the different main objectives of the project, the design and development of this second application has the following particular objectives:

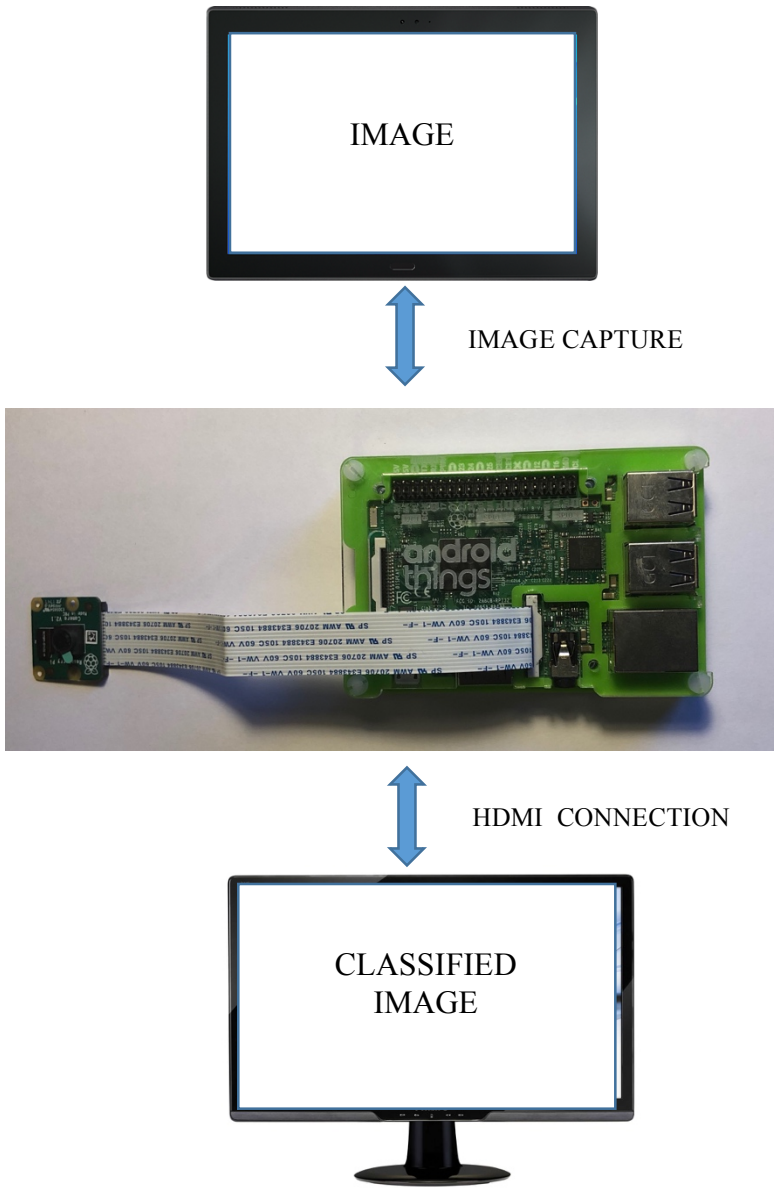
- Increase knowledge of the handling of the different GPIO pins of the Raspberry Pi.
- Acquire the necessary knowledge for the use of the Android camera API.
- Learn to integrate the TensorFlow library in different Android projects.
- Introduction to data feeding in a TensorFlow model

At the beginning of this application we also make use of an example provided by Android through its official website. Modifications are made to this example both in the handling of peripherals (display, camera, etc.) and in the output format of the results. In order to carry out its development, the following fundamental requirements must be taken into account:

- Raspberry Pi 3
- Rainbow HAT
- Official Camera Board V2 to Raspberry Pi
- Android Studio 3.0 or higher
- HDMI screen (optional)

As for the design of this second project, it consists of an application for the Raspberry where is defined the use of a GPIO pin corresponding to a button, which will generate a KeyEvent subsequently processed to generate the action of taking the photo and the different functions for the initialization and loading of the classifier model.

Throughout this section the structure and the different parts and most important functions of this image classification system based on TensorFlow are exposed. The following image shows the scheme of the application.



1.5 Conclusions

After carrying out the research necessary for the development of this project and the implementations of the different applications, several conclusions have been reached. Some of these are highlighted below.

On one hand, thanks to the search on the past and present of IoT technologies and the new operating system of this technology from Google, there is a clear future of the devices that make up this concept . This future is characterized by an exponential increase of these devices with connected sensors and therefore of a massive digital data growth.

Thanks to the developments with this new technology, Android Things, together with the other technologies, has demonstrated the numerous functions that can be performed within this field and the innumerable of possible applications still to be developed.

In general, regarding the initial objectives, these are successful. By making and obtaining the resulting systems that make up the final applications, we reach the main goal of serving those applications as a start for future developments to improve our lives. Throughout the realization of the project, the partial objectives initially defined have also been achieved. Regarding the scope of these partial objectives it should be noted that thanks to the development of the project, a good knowledge of the Android Things operating system has been acquired.

1.6 Future improvements

As mentioned, throughout the realization of the project , we have worked with innovative technologies in constant updated, numerous ideas and improvements as well as different developments have arisen in relation to the different applications designed. These ideas have increased with the communication established among numerous users of these technologies through the Google communities. These future ideas have been classified into two aspects: future improvements for the different applications developed and future improvements through the technologies used.

- Future improvements of the applications:

    - Android Things application with Firebase. Respect to this application, there are countless possible improvements to be implemented in the future. Starting with the mobile application you can design more complex interface with more extensive control over a greater number of peripherals of the Raspberry. The next step of the complete control system, it would be its integration with any production environment such as a boiler in a home or a home automation system.

    - Android Things application together with Tensorflow. Regarding this project, there are numerous fields for its application in different production environments. According to these, you can develop a series of improvements focused on your objectives. One of these examples would be its integration in the field of medicine, helping blind people. Some improvements for this case would be the integration of a speaker as a peripheral or a partial integration with the project of Android Things with Firebase to capture images remotely.

- Future improvements through the technology used:

With respect to Firebase, apart from the RealTime Database service used in the application, this technology offers other services with the possibility of developing numerous cases of uses. Some of these possible use cases are the following.

  - Design a satisfactory incorporation flow. With the use of the Firebase Authentication and Remote Config services, you can design a remote control application that can be accessed through a Facebook, Twitter, Google account .
  - Customize a welcome screen for each user through the Remote Config and Analytics services.
  - The design of a system to follow the path of users through different devices to obtain more data and understand better to user. The Big Query and Analytics services are used for this improvement.

Finally, another thing that can be done with Tensorflow technology is the re-training of a classification model of a data sheet, join to the operating system of the internet of things, Android Things, to design a more personalized classification and detection system. This training is made from the parameters of a modern recognition model that reuses the characteristics extraction capabilities of itself.